СПЕЦИАЛИЗИРАН НАУЧЕН СЪВЕТ
ПО ИНФОРМАТИКА И ПРИЛОЖНА МАТЕМАТИКА ПРИ ВАК

Евгений Христов Николов

# ИЗСЛЕДВАНИЯ ВЪРХУ ИНФОРМАЦИОННАТА СИГУРНОСТ НА КОМПЮТЪРНИ ЗАЩИТНИ СИСТЕМИ, РАБОТЕЩИ В МРЕЖОВА СРЕДА И ПРОТОКОЛИ TCP/IP

Научна специалност
01.01.12 Информатика

АВТОРЕФЕРАТ

на дисертация за присъждане на научната степен
"Доктор на математическите науките"

СОФИЯ

2001

Съдържание Стр.



Дисертацията е обсъдена и насочена за защита на разширено заседание на секция "Софтуерни технологии" на Институт по математика и информатика при БАН.

Защитата ще се състои на ..... от ..... ч. в зала ..... на Института по математика и информатика при БАН на разширено заседание на Специализирания научен съвет по информатика и приложна математика при ВАК. Материалите по защитата са на разположение на интересуващите се в библиотеката на Института по математика и информатика при БАН.

РЕЗЮМЕ

ЦЕЛ. Изследване на информационната сигурност на компютърни защитни системи (CPS), работещи в мрежова среда и протоколи TCP/IP. По отношение на проектирането и оптимизацията се изследва влиянието на състава, структурата и управлението на CPS върху характеристиките им и създаване на система от критерии за оценка. По отношение на използването на CPS се изследва влиянието на решаваните от CPS задачи върху характеристиките, състава, структурата и управлението им.

ЗАДАЧИ. Създаване на обща методология за проектиране и оптимизация на CPS чрез аналитично и симулационно моделиране с извършване на оптимизационен процес по определени критерии за оценка. Разработване и прилагане на програмни модели за изследване характеристиките на CPS по отношение на тяхното използване. Извършване на планирани симулационни експерименти. Формулиране на изводи и препоръки на базата на получените резултати.

АКТУАЛНОСТ. CPS в своето развитие през последните години се характеризират с рязко усложняване и използване на различни софтуерни и хардуерни средства. Това превърна обикновените компютърни антивирусни програми от средство за еднократна локална проверка на определени зони от оперативната памет в самостоятелна система със сложни функции, гарантиращи данновата, компютърната, комуникационната и информационната сигурност в мрежова среда и протоколи TCP/IP (Интернет/Екстранет/Интранет). Поради това изследването на информационната сигурност на CPS се превръща в сложна задача, при която е наложително използването на моделиране и оптимизация.

КЛАСИФИКАЦИЯ. В момента съществуват повече от 300 вида CPS на основата на повече от 80 вида процесори и на повече от 120 вида защитни технологии. Предложена е следната елементарна класификация, обхващаща трите основни функции на CPS: даннова, компютърна и комуникационна защита. Всяка има по четири степени на изменение, съответно за данновата защита - на ниво бит, байт, сегмент и пакет; за компютърната защита - на ниво сменяеми и постоянни носители, клавиатура и локална операционна система; за комуникационната защита - на ниво идентификационни, транспортни и мрежови протоколи, и мрежова операционна система.

ТЕРМИНИ. За да има яснота в използването им, в дисертацията са разгледани накратко някои утвърдили се вече термини и понятия като: информационна, даннова, компютърна и комуникационна сигурност, компютърна защитна система, мрежова среда, протоколи TCP/IP, информационна среда, компютърен вирус и т.н. Цитирането на литературата е направено в квадратни скоби [...], а на публикациите - в двойни квадратни скоби [[...]].

СТРУКТУРА. Дисертацията има резюме, въведение, четири глави, заключение, публикации, благодарности и литература. Тя съдържа 194 страници с текст, 144 страници с графики, 24 страници с таблици, 40 страници с фигури, 291 цитирани литератури, 294 съкращения и означения, 382 номерирани изрази, 840 графики, 28 таблици, 14 фигури и 25 блокови схеми.

ПРИНОСИ. 1) Направен е преглед на известните методи, модели и проблеми с отношение към задачата за оптимално проектиране на CPS, като са посочени техните предимства и недостатъци и е предложена класификация на CPS, основаваща се на три основни елемента: даннова, компютърна и комуникационна защита; 2) Формулирано е решение на задачата за оптимално проектиране на CPS чрез аналитично и симулационно моделиране, при което, при зададени тип на изпълняваните функции и задачи и тип на структурата на CPS се извършва оценка на нейните характеристики, като в съответствие с това се прилага оптимизационен процес; 3) Дефинирани са еталонни структури, имащи различна организация на паметта и на връзките процесор - памет, различен брой еднотипни копроцесори и функционално специализирани копроцесори; 4) Дефинирани са критерии за оценка на CPS, разделени на общи – време за обработка, скорост за обработка, цена за обработка и специфични - средна и относителна продуктивност, реална, относителна и обща производителност, реална, относителна и обща заетост, като са определени съотношенията между тях в зависимост от параметрите на избраните структури; 5) Разработени са алгоритми и програмни модели за изследване на основните характеристики на CPS и са избрани източниците на случайни величини и вероятностните функции; 6) Създаден е програмен пакет за симулационно моделиране на CPS, състоящ се от отделни симулационни програми за всяка от избраните структури; 7) Изследвано е влиянието на структурата, състава и управлението на CPS върху нейните характеристики при зададен тип задачи и влиянието на типа задачи върху характеристиките на CPS при зададени структура, състав и управление; 8) Постигнатите практическите резултати, са използвани в проекти на Национална лаборатория по компютърна вирусология с възложител Българска академия на науките и Националния фонд за научни изследвания.

# ГЛАВА ПЪРВА

Изследването на CPS може да се раздели на три класа задачи: за проектиране, избор и усъвършенстване. В този случай съществени са задачите за проектиране. За тяхното решаване съществуват два основни подхода: на моделирането и на измерването. Явно е, че вторият има приложение в случаите, когато е налице реален обект, който е прототип на търсената система. Интерес за нас ще представлява моделирането, тъй като в повечето от случаите на проектиране няма реален обект, който да измерваме. Моделиращият подход се реализира чрез две основни форми: аналитично и симулационно моделиране, които ще бъдат предмет на разглеждане в по-нататъшното изложение. Освен това, при решаването на задачи за проектиране се използват два основни метода: методът на итеративното и на йерархичното проектиране. Итеративното проектиране се състои в намирането на първоначално решение на проблема и преобразуването му в приемливо според някакви изисквания решение чрез последователност от итеративни изменения, т.е. това е процес, при който с последователни приближения се стига до едно крайно решение. Именно тук - в определянето на първоначалната конфигурация на CPS може да се търси мястото на аналитичното моделиране, тъй като елементарни аналитични модели дават разумни резултати [10,13], [[12]]. Аналитичните модели се използват и при проверката дали първоначалната конфигурация удовлетворява спецификацията за проектиране, както и в определяне на конкретните изменения в първоначалната конфигурация. При итеративната процедура в случай на проектиране на CPS с най-голям успех се използват вероятностни модели (Марковските вериги и опашковите модели) [7, 12]. Йерархичното проектиране се основава на противоположен принцип, т.е. съставът и структурата на CPS, както и характеристиките на техните компоненти, се извеждат непосредствено от проектните спецификации по такъв начин, че резултатът да се счита за функционално коректен [18, 33]. Този вид проектиране е възможен само в някои елементарни случаи и неговото приложение в проектирането на CPS е ограничено. Той се използва основно при проектирането на големи и сложни системи, където е възможно и удобно да се приложи неговата основна идея: йерархично разпределени нива на абстракции със съответна степен на опростяване, които се включват една в друга [29, 46]. Като най-подходяща стратегия на проектиране на CPS може да се избере комбинацията, при която първоначалният проект се реализира с аналитичен модел (най-често вероятностен), към който се прибавят опитът и интуицията на проектанта [44, 71]. След това първоначалният проект се реализира като симулационен модел, който итеративно се изменя, докато не се удовлетворят всички изисквания. Задачите за проектиране са тясно свързани със задачите за оптимизация и често те се обединяват под името задачи за оптимално проектиране [82, 97]. Оптимално проектиране може да се опише като процес, при който се осъществява проект за дадена система, като при това определена функция $F(\vec{X}) = F(X_1, X_2, ..., X_n)$, (1.1) приема екстремална стойност $F(\vec{X}) = F(\vec{X})_{extr}$ (1.2) където: $X_1, X_2, ..., X_n$ (елементи на вектора $\vec{X}$) са параметри, които определят функционалните характеристики на обекта. Функцията $F(\vec{X})$ се нарича целева функция на оптимизационен процес. При определяне на екстремума на целевата функция трябва да бъдат спазвани определени ограничителни условия, наложени от физическата страна на процесите в обекта или удовлетворяването на други функции, свързани с него [64, 68]. Така пълното формулиране на задачата за оптимално проектиране е следното: търси се екстремум на целева функция от вида:

$$F(\vec{X}) = F(X_1, X_2, ..., X_n)_{extr} \quad (1.3)$$ при ограничението $$F_r(\vec{X}) = F_r(X_1, X_2, ..., X_n) \geq K_r \rightarrow r = 1, 2, ..., n \quad (1.4)$$

където $K_r$ са константи или нули. Функциите $F(\vec{X})$ и $F_r(\vec{X})$ могат да бъдат алгебрични полиноми, диференциални или интегрални уравнения и други. Главните условия, на които трябва да отговарят, са да бъдат известни и да описват достатъчно точно процесите в проектирания обект. Задачите за оптимално проектиране се решават с помощта на два основни типа методи: аналитични и числени методи за оптимизация [79, 98]. За оптималното проектиране на CPS е най-добре да се използва нелинейното програмиране, което се характеризира с най-голяма универсалност и приспособимост към изискванията на аналитичното моделиране. В заключение на разглежданите въпроси, може да се направи изводът, че задачата за оптимално проектиране на CPS се вписва в проблематиката на актуалните научни изследвания и за нейното решаване имаме достатъчно ефективни средства.

ГЛАВА ВТОРА

*Аналитичен модел.* CPS може да се представи чрез определено множество от съставни елементи (CE). За нашите цели може да се приеме, с разумен компромис между реалност и идеализиране, че броят на основните типове съставни елементи е десет. Това може да се изрази символично по следния начин: {CPS} := {FT} {PO} {SG} {NC} {PC} {HT} {ST} {DT} {PH} {PS} (2.7) където: CPS – Computer Protection System, FT – Functions and Tasks, PO – Processes and Operations, SG – Segments and Groups, NC – Network Controls, PC – Protocol Controls, HT – Hardware Tools, ST – Software Tools, DT – Dialog Tools, PH – Peripheral Hardware, PS – Peripheral Software. Всеки тип съставен елемент може да бъде представен чрез своите типове характерни компоненти (TC):

{FT} := {InF} {OuF} {InT} {OuT} (2.8a) {PO} := {InP} {OuP} {InO} {OuO} (2.8b)
{SG} := {InS} {OuS} {InG {OuG} (2.8c) {NC} := {NOS} {NAP} {NLP} {NDP} (2.8d)
{PC} := {TCP} {IP} {UDP} {ICMP} (2.8e) {HT} := {IPS} {DAS} {DPS} {EMS} (2.8f)
{ST} := {LOS} {LAP} {LLP} {LDP} (2.8g) {DT} := {SDS} {VDS} {CDS} {BDS} (2.8h)
{PH} := {InD} {OuD} {IOD} {TCD} (2.8i ) {PS} := {POS} {PAP} {PLP} {PDP} (2.8j )

където: InF - Input Functions, OuF – Output Functions, InT – Input Tasks, OuT – Output Tasks, InP – Input Processes, OuP – Output Processes, InO – Input Operations, OuO – Output Operations, InS – Input Segments, OuS – Output Segments, InG – Input Groups, OuG – Output Groups, NOS – Network Operating Systems, NAP – Network Application Programs, NLP – Network Library Programs, NDP – Network Driver Programs, TCP - Transmission Control Protocol, IP – Internet Protocol, UDP - User Datagram Protocol, ICMP – Internet Control Message Protocol, IPS – Integrated Processor Systems, DAS – Discreet Active Systems, DPS - Discreet Passive Systems, EMS – Electro Mechanical Systems, LOS – Local Operating Systems, LAP – Local Application Programs, LLP - Local Library Programs, LDP – Local Driver Programs, SDS – Screen Dialog Systems, VDS – Voice Dialog Systems, CDS – Cart Dialog Systems, BDS – Biometric Dialog Systems, InD – Input Devices, OuD – Output Devices, IoD – Input Output Devices, TCD – Transform Convert Devices, POS - Peripheral Operating System, PAP - Peripheral Application Programs, PLP - Peripheral Library Programs, PDP - Peripheral Driver Programs [[4, 5, 12]]. Заедно с това, CPS може да бъде представена като една обща матрица ($MAT^{CPS}$=$\mathbf{M}^{CPS}$), включваща в себе си матриците на типовете съставни елементи. Процесът на формализираното описание може да се детайлизира, като матриците на типовете съставни елементи се разложат на подматрици, представящи типовете характерни компоненти. Типовете характерни компоненти могат да се опишат формално чрез подниво на подматриците (*под*подматрици), като новото представяне включва квадратни матрици, в които участва следващо подниво на дефинирането, наречено градивни компоненти (CC). Матриците отразяват връзките между всички CC в даден вариант на свързване, като се използва 1 при свързване и 0 при отсъствие. При тези предположения матрицата, отразяваща даден вариант на свързване на отделните CC (за примера, те са 16 на брой и са номерирани шестнадесетично, съответно от 0 до F), ще има минимално свързване, показано чрез (2.21) и максимално свързване, показано чрез (2.22).

$\mathbf{M}_{min}^{xxx}$ = (2.21)

| 0 1 2 3 4 5 6 7 8 9 A B C D E F | CC |
|---|---|
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 | **0** |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **1** |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **2** |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **3** |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **4** |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **5** |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **6** |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **7** |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **8** |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **9** |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **A** |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **B** |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **C** |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **D** |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **E** |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **F** |

$\mathbf{M}_{max}^{xxx}$ = (2.22)

| 0 1 2 3 4 5 6 7 8 9 A B C D E F | CC |
|---|---|
| 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 | **0** |
| 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 | **1** |
| 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 | **2** |
| 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 | **3** |
| 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 | **4** |
| 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 | **5** |
| 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 | **6** |
| 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 | **7** |
| 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 | **8** |
| 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 | **9** |
| 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 | **A** |
| 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 | **B** |
| 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 | **C** |
| 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 | **D** |
| 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 | **E** |
| 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 | **F** |

*Видове методики.* Съгласно направените разглеждания в първа глава и предложената концепция за аналитичен модел на CPS ще разделим проблемите на проектиране на CPS чрез аналитични средства на три [5, 11, 23, 40, 42, 49], [[4, 5, 10, 11, 12]]. Всяка част от тях ще бъде обобщена под формата на методика за оптимизационен процес, имайки предвид, че процесът на проектиране е итеративна процедура за намиране на нужното решение. Като най-важни са отделени проблемите, свързани със състава, структурата и управлението на CPS, като на тази основа ще се разгледат последователно три методики на оптимизационен процес. Тяхното подреждане като последователност от действия не трябва да се разглежда като нещо задължително, а като отражение на класическия подход при проектирането, при който се върви от частното към общото и от простото към сложното [50], [[12]]. Всяка методика ще включва извеждане на аналитични изрази, обобщаващи основните параметри на оптимизационния процес в решима форма, определяне на аналитичните условия, съобразно които ще се осъществява оптимизационния процес в съответствие с избрани критерии, и обобщение на всичко под формата на алгоритъм за програмна реализация, която е осъществена със средствата на универсалния алгоритмичен език PASCAL/DELPHI. Като база за програмните експерименти е използвана компютърната техника в Национална Лаборатория по Компютърна Вирусология при БАН.

МЕТОДИКА НА ОПТИМИЗАЦИОНЕН ПРОЦЕС ЗА СЪСТАВА. Проектирането на CPS, в съответствие с направените вече разглеждания, може да се представи като процес, при който трябва да се избере състав и структура на СС по такъв начин, че да се удовлетворяват изискванията за скорост, обем и цена. Тук ще разгледаме проблемите за избор на оптимален състав на CPS, а по-нататък – проблемите за избор на оптимална структура на CPS, следвайки принципа на подоптимизацията.

164. *Аналитични изрази.* Нека е дадено някакво множество от СС, образуващи една система от модули: $m_1, m_2, m_3, ..., m_j, ..., m_z$ (2.23). Системата може да има различни състояния, множеството от които може да се раздели най-общо на три групи: 1) Първа група – в системата работи поне един от СС, тогава общият брой на състоянията от групата е: $H_{com}=2^M-1$ (2.24) където: $M=\sum_{j=1}^{z} m_j$ (2.25); 2) Втора група – в системата работят *i* на брой СС, тогава общият брой на състоянията от групата е: $H_i=(G_M)^i$ (2.26) или $H_{com}=\sum_{i=1}^{M} H_i$ (2.27); 3) Трета група – в системата работят комбинации от СС, които ще означаваме като $r_{i,v}$ при което състоянията са: $H_{r_{i,v}}=\prod_{j=1}^{z}(G_{m_j})^{u_j}$ (2.28) и $H_i=\sum_{v=1}^{(W_i)^c} H_{r_{i,v}}$ (2.29), където броят на събираемите в (2.29) е обвързан с условието $(W_i)^c=\begin{bmatrix}(G_{i-(z-1)})^i \rightarrow i \le 0{,}5M\\(G_{i-(z-1)})^{z-1} \rightarrow i > 0{,}5M\end{bmatrix}$ (2.30). Всяка комбинация $r_{i,v}$ може да се изрази като $r_{i,v}=u_1, u_2, ..., u_j, ..., u_z$ (2.31) при което е в сила равенството $\sum_{j=1}^{z} u_j=i$ (2.32) и всяка комбинация $r_{i,v}$ се отличава по своята група от стойности за $u_j$, които се изменят от 0 до i. Общият брой неповтарящи се комбинации $r_{i,v}$: $\sum_{i=1}^{M}((W_i)^c-q_i)$ (2.33), където $q_i$ е броят на невъзможните стойности за $r_{i,v}$, а невъзможна стойност за $r_{i,v}$ ще има, ако $u_j>m_j$. Тогава $(G_{m_j})^{u_j}=0$ (2.34). По отношение на основната характеристика на състоянието (ВС) на комбинацията $r_{i,l}$ можем да запишем $BC_{r_{i,l}}=\sum_{j=1}^{z} u_j bc_j$ (2.35), където l е пореден номер на комбинацията $r_{i,v}>0$. Следователно: $PRB_{r_{i,l}}^{work}=H_{r_{i,l}}\prod_{j=1}^{z}(prb_j^{work})^{u_j}(prb_j^{denial})^{m_j-u_j}=\prod_{j=1}^{z}(G_{m_j})^{u_j}(prb_j^{work})^{u_j}(prb_j^{denial})^{m_j-u_j}$ (2.36). Нека сега да разделим на (P+1) групи комбинацията $r_{i,l}$, като имаме предвид следните признаци: 1) Група '1', за която: $BC_{r_{i,l}} \ge \sum_{p=1}^{P} BC_p$ (2.37a); 2) Група '2', за която: $BC_{r_{i,l}} \ge \sum_{p=1}^{P-1} BC_p$ (2.37b); 3) Група 'f', за която: $BC_{r_{i,l}} \ge \sum_{p=1}^{P+(f-1)} BC_p$ (2.37c); 4)

Група 'P', за която: $BC_{r_{f1}} \geq BC_1$ (2.37d); 5) Група 'P+1', за която: $BC_{r_{f1}} < BC_1$ (2.37e). Тогава можем да запишем следния израз за вероятността за решаване на задачи, влизащи в групата 'f': $PRB_f^{work} = \sum_{(f)} PRB_{r_{f1}}^{work}$ (2.38), където сумирането се осъществява по всички $r_{f1}$, влизащи в състава на групата 'f'. В крайна сметка, изразът за $PRB_p^{work}$ ще има вида: $PRB_p^{work} = \sum_{f=1}^{P+(p-1)} \sum_{(f)} \prod_{j=1}^{z} (G_{m_j})^{u_j} (prb_j^{work})^{u_j} (prb_j^{denial})^{m_j - u_j}$ (2.39), където сумирането в областта на 'f' се осъществява при спазване на неравенството: $BC_{r_{f1}} \geq \sum_{p=1}^{P+(f-1)} BC_p$ (2.40). Когато в системата участвуват само еднородни СС, т.е. z=1, вземайки предвид (2.40), можем да опростим $PRB_p^{work} = \sum_{i=m_p}^{M} (G_M)^i (prb^{work})^i (prb^{denial})^{M-i}$ (2.41) или $PRB_p^{work} = \sum_{i=m_p}^{M} (-1)^{i-m_p} (G_M)^i (G_{i-1})^{m_p-1} (prb^{work})^i$ (2.42), където $m_p$ е закръгленото към по-голямо цяло число частно $\sum_{i=1}^{p} \frac{BC_i}{bc}$, в което $BC_i$ е основна характеристика на функция и задача с i-ти приоритет, а bc е аналогична основна характеристика за един СС (модул от системата) от j-ти тип.

*Аналитични условия.* Позовавайки се на направените по-горе разглеждания, е възможно да бъдат определени условия, съобразно които да се оптимизира съставът на CPS по отношение на нейните СС. Аналитичният вид на условията не е сложен и няма нужда от особени доказателства и извеждания, така че можем да запишем: $\sum_{j=1}^{z} m_j bc_j \geq \sum_{p=1}^{P} BC_p$ (2.43) $\sum_{j=1}^{z} m_j scp_j = MIN$ или $\sum_{j=1}^{z} m_j pri_j = MIN$ (2.44) $PRB_j^{work} \geq \sqrt[s]{PRB_{g.p}^{work}}$ (2.45) където scp е обем/размер и pri е цена на j-тия СС, а $PRB_{g.p}^{work}$ е зададената вероятност за безотказна работа на системата за време t при решаване на задача с p-ти приоритет и s е броят на наименованията на основните СС (модули) в системата. Както беше посочено вече, СС могат да бъдат дублирани (пълно или частично). Условията (2.43) и (2.44) са еднакви и за двата случая, но условието (2.45) е различно за пълно или частично дублираните СС и може да се сведе, както ще бъде показано, съответно до задача от нелинейното и до задача от линейното програмиране. Анализът е следният: 1) Частично дублирани СС – Съгласно [185, 206, 211] можем да запишем израз, определящ минимален количествен състав на CPS при еднородни СС от тип j и при изпълнение на функции и задачи с p-ти приоритет, като се осигурява зададената стойност на $PRB_p^{work}$, която ще има вида: $PRB_p^{work} = 1-(1-prb_j^{work})^{m_j}$ (2.46). От израза (2.46), като се вземат предвид (2.43) и (2.45) ще получим: $m_j \geq \frac{BC_p \ln(1-\sqrt[s]{PRB_{g.p}^{work}})}{bc_j \ln(1-prb_j^{work})}$ (2.47). Ако обозначим: $b_{pj} = \frac{\ln(1-\sqrt[s]{PRB_{g.p}^{work}})}{\ln(1-prb_p^{work})}$ (2.48), където $b_{pj}$ е цяло число, закръглено към по-голямо, имаме основание да заменим условията (2.43) и (2.45) с едно условие: $\sum_{j=1}^{z} m_j bc_j \geq \sum_{j=1}^{z} \sum_{p=1}^{P} b_{pj} BC_p$ (2.49); 2) Пълно дублирани СС – в този случай, като вземем предвид (2.39) за условието (2.45) можем да запишем: $\sum_{f=1}^{P-(p+1)} \sum_{(f)} \prod_{j=1}^{z} (G_{m_j})^{u_j} (prb_j^{work})^{u_j} (prb_j^{denial})^{m_j-u_j} \geq \sqrt[s]{PRB_{g.p}^{work}}$ (2.50). Ако направим следните заместванията: $m_j = x_j$, z=n, $scp_j$ или $pri_j$ с $a_i$, $bc_j = c_i$ и $\sum_{j=1}^{z} \sum_{p=1}^{P} b_{pj} BC_p = C$, е възможно (2.49) да бъде записано в следния вид: $\begin{bmatrix} a_1x_1 + a_2x_2 + ... + a_nx_n = min \\ c_1x_1 + c_2x_2 + ... + c_nx_n = C \\ x_1 \geq 0, x_2 \geq 0, ..., x_n \geq 0 \end{bmatrix}$ (2.51), където $x_1, x_2, ..., x_n$ са цели числа. Това вече е типичен запис за задача от линейното програмиране и методите за нейното решаване са добре известни [236, 272]. Ако използваме направените вече означения и като прибавим към тях

следните полагания: $u_j=y_i$, $\sum_{p=1}^{P} BC_p=C$, $\sqrt[s]{PRB_{g.p}^{work}}=PRB_j^{work}$ и $P=J$, то условието (2.50) ще има вида:

$$\begin{bmatrix} a_1x_1+a_2x_2+...+a_nx_n=\min \\ \sum_{j=1}^{J}\sum_{(i)}\prod_{(i)}(G_{x_i})^{y_i}(prb_i^{work})^{y_i}(prb_i^{denial})^{x_i-y_i} \ge PRB_1^{work} \\ \sum_{j=1}^{J-1}\sum_{(i)}\prod_{(i)}(G_{x_i})^{y_i}(prb_i^{work})^{y_i}(prb_i^{denial})^{x_i-y_i} \ge PRB_2^{work} \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ \sum_{(i)}\prod_{(i)}(G_{x_i})^{y_i}(prb_i^{work})^{y_i}(prb_i^{denial})^{x_i-y_i} \ge PRB_J^{work} \end{bmatrix} \quad (2.52),$$

където $x_1 \ge 0, x_2 \ge 0, ..., x_n \ge 0$ и $x_1, x_2, ..., x_n$ са цели числа, а $y_i$ е някаква функция на $x_i$ и $i$. По такъв начин получихме формулиране на проблема, което е типично за нелинейното програмиране и може да се реши със съответните методи [6, 28, 35, 45, 62, 83, 94, 95, 107, 131]. Нека сега да се спрем на основните стъпки, от които се състои методиката на оптимизационен процес за състава на CPS.

*Алгоритъм на оптимизационен процес*. Първо ще въведем още едно понятие $a_{pj}$ - <u>специфична цена</u> или <u>специфичен обем</u> на единица основна характеристика, като $a_{pj}=\frac{scp_j}{bc_j}\left[\frac{\ln(1-\sqrt[s]{PRB_{g.p}^{work}}}{\ln(1-prb_j^{work})}\right]$ (2.53) или $a_{pj}=\frac{pri_j}{bc_j}\left[\frac{\ln(1-\sqrt[s]{PRB_{g.p}^{work}}}{\ln(1-prb_j^{work})}\right]$ (2.54). За целите на оптимизационния процес за състава на CPS, е по-удобен изразът (2.54). Освен това, имайки предвид (2.48), ще се уговорим елементите на матрицата $b_{pj}$ да се закръглят към по-голямо цяло число. Величината $a_{pj}$ има смисъл на матрица с $p$ стълба и $j$ реда. Вариантите на алгоритъма са:

<u>Частично дублирани СС</u> – основните стъпки са: 1) Изчисляване на елементите на матрицата $b_{pj}$ чрез (2.48); 2) Изчисляване на елементите на матрицата $a_{pj}$ чрез (2.54); 3) Определяне на минималните елементи на $a_{pj}$ по стълбове; 4) Определяне на съответните им редове; 5) Определяне на оптимален тип СС за съответния приоритет в съответствие с определените редове на $a_{pj}$ в стъпка 3; 6) Изчисляване броя на СС от $j$-ти тип чрез $m_j=\frac{BC_p b_{pj}}{bc_j}$ (2.55) при което индексите на матрицата $b_{pj}$ са същите, които бяха определени в стъпки 2 и 3; 7) Закръгляване на получената стойност за $m_j$ към по-голямото цяло число; 8) Определяне на търсения състав от СС в първо приближение; 9) Изчисляване на $\Delta BC$ чрез (2.56) $\sum_{j=1}^{z} m_j bc_j - \sum_{p=1}^{P} BC_p b_{pj} = \Delta BC$ (2.56); 10) Определяне на <u>двойки основни характеристики</u> (PBC), съответно на $j$-тия тип СС и на $(j+v)$-тия тип СС, чиято разлика, след като се сумира, клони към $\Delta BC$, т.е. в сила е (2.57), $\Delta BC-\sum_{j=1}^{z}\left(bc_j-bc_{j+v}\right) \ge 0=\min$ (2.57), при което процесът на определяне се ръководи от следните по-важни съображения: 1. Сумирането да обхваща всички двойки характеристики; 2. Сумата от разликите трябва да клони максимално плътно към $\Delta BC$, защото така се получава максимум на характеристиките за състава на CPS; 3. Определянето на $v$ трябва да се ръководи от изискването: $v \le (z-j)$, при това обикновено колкото е по-голямо $v$, толкова е по-оптимален нашият състав; 4. Броят на двойките основни характеристики за една стойност на $j$ не трябва да превишава стойността на $m_j$; 5. Заместването на $j$ с $j+v$ трябва да се ръководи от изискването $prb_{j+v} \ge prb_j$; 11) Заместване на $m_j$ в определения в стъпка 8 състав в първо приближение с $m_{j+v}$, при което, ако в състава се получат $g$

еднакви разлики на основни характеристики (разгледани в стъпка 10), то съответните $m_j$ ще се заместят с $(m\text{-}g)_j$ и освен това ще се прибавят $g_{j+v}$; 12) Фиксиране на получения в стъпка 11 състав от $m_j$ като крайно решение на задачата за оптимизиране на състава на СС в случай на частично дублиране.

Пълно дублирани СС – основните стъпки са: 1) Изчисляване на елементите на матрицата $b_{ij}$ чрез (2.48); 2) Изчисляване на елементите на матрицата $a_{ij}$ чрез (2.54); 3) Определяне на минималните елементи на $a_{ij}$ по стълбове; 4) Определяне на съответните им редове; 5) Определяне на оптималния тип СС за съответния приоритет в съответствие с определените редове на $a_{ij}$ в стъпка 4; 6) Изчисляване на броя на СС от j-ти тип чрез (2.55); 7) Изчисляване на $PRB_p^{work}$ чрез (2.39); 8) Проверка дали $PRB_p^{work}$ клони към $PRB_{g.p}^{work}$ (зададено); 9) Ако разликата е пренебрежима, то намереният в стъпка 6 състав е търсеният в първо приближение; 10) Ако разликата е незадоволителна, увеличаваме количеството на оптималния тип СС за съответния приоритет с единица; 11) Изчисляваме новите стойности на $PRB_p^{work}$ (2.39); 12) Проверяваме, дали $PRB_p^{work}$ клони към $PRB_{g.p}^{work}$ (зададено); 13) Ако новата разлика е пренебрежима, то коригираният състав е търсеният в първо приближение; 14) Ако новата разлика е незадоволителна, повтаряме стъпки 10, 11, 12; 15) Изчисляване на $\Delta BC$ чрез (2.56); 16) Определяне на двойки основни характеристики съответно за j-тия тип СС и за (j+v)-тия тип СС, като имаме предвид (2.57) и съображенията, изложени в стъпка 10 на алгоритъма на оптимизационен процес за състава на частично дублирани СС; 17) Заместване на $m_j$ с цел намаляване на $\Delta BC$ в определения в стъпки 9 или 13 състав в първо приближение с нова стойност $m_{j+v}$, при което имаме предвид съображенията, изложени в стъпка 11 на алгоритъма на оптимизационен процес за състава на частично дублирани СС, и следните нови уговорки: 1. Стойността v може да бъде положителна или отрицателна; 2. След всяка поредна замяна на СС от j-ти тип с СС от (j+v)-ти тип трябва да се прави проверка на стойностите на $PRB_p^{work}$; 3. Заместването е възможно, само при $PRB_p^{work} \geq \sqrt[s]{PRB_{g.p}^{work}}$ (2.58); 18) Фиксиране на получения в стъпка 17 коригиран състав от $m_j$ като крайно решение на задачата за оптимизиране на състава на СС в случай на пълно дублиране.

МЕТОДИКА НА ОПТИМИЗАЦИОНЕН ПРОЦЕС ЗА СТРУКТУРАТА. Следващ етап в проектирането на CPS е оптимизирането на структурата ѝ в съответствие с избрани критерии за оптимизация. Основен момент в този етап е видът на суперпозицията на ефективностите на отделните функции и задачи - линейна и нелинейна, така че оптимизационният процес за структурата на CPS ще се разглежда в съответните два основни случая. Като изходни данни трябва да бъдат зададени: 1) $CON^{CPS}$ – матрицата, описваща варианта на връзките между отделните СС; 2) $DIS^{CPS}$ – матрицата, описваща варианта на разпределението на отделните функции и задачи по съответните СС; 3) Брой и състав на основните и неосновните СС; 4) Характеристиките на отделните функции и задачи в таблична форма; 5) Характеристиките на отделните СС в таблична форма.

*Аналитични изрази.* Ще разгледаме аналитични изрази за $EFI^{CPS}$ и $PRB_{n.k}^{sol}$ в двата основни случая.

Линейна суперпозиция на ефективностите. Позовавайки се на [148, 164], можем да запишем: $EFI^{CPS} = \sum_{k=1}^{K} EFI_k^{FT} \prod_{n=1}^{S} PRB_{n.k}^{sol}$ (2.59), където: $EFI^{CPS}$ е ефективността на CPS; $EFI_k^{FT}$ е ефективността на k-тата функция и задача; $PRB_{n.k}^{sol}$ е вероятността за решение на k-тата функция и задача от СС с n-тото функционално название. Вижда се, че при определяне на вероятността е необходимо да се отчита не общото количество СС в системата, а само СС, които участвуват в решаването на k-тата задача. При това е възможно да съществуват два случая: 1) Функцията и задачата се решава само с един СС – има две възможности: 1. Еднородни СС: нека да има m еднородни СС, участвуващи в решаването на k-тата задача и всеки СС да има надеждност, която е $prb^{work}$. Тогава вероятността за i-тото състояние на СС (където i е броят на работещите СС) ще бъде: $PRB_i^{work} = (G_m)^i (prb^{work})^i (prb^{denial})^{m-i}$ (2.60) и следователно $PRB_{n.k}^{solv} = \sum_{i=1}^{m} (G_m)^i (prb^{work})^i (prb^{denial})^{m-i}$ (2.61) 2. Неоднородни СС: нека да имаме някакво количество СС,

чиито брой е: $M=\sum_{j=0}^{L} m_j$ (2.62) където: $L$ е броят на различните типове СС, участващи в решаването на $k$-тата задача; $m_j$ – броят на СС от $j$-ти тип. Надеждността на СС от $j$-ти тип се характеризира с величината $prb_j^{work}$. Тогава вероятността за това, че няма едновременно работещи СС ще бъде: $\prod_{j=1}^{L}(1-prb_j^{work})^{m_j}=\prod_{j=1}^{L}(prb_j^{denial})^{m_j}$ (2.63) и следователно $PRB_{n,k}^{solv}=1-\prod_{j=1}^{L}(prb_j^{denial})^{m_j}$ (2.64). 2) Функцията и задачата се решава с няколко СС – има две възможности: 1. Еднородни СС: нека $k$-тата функция и задача да бъде разделена на $R$ отделни части, които ще се изпълняват в отделните СС; нека да означим комбинациите (на брой $2^M-1$) от суми на отделни части по следния начин: $\sum_{i=1}^{x} i=Q$ (2.65). Тогава, използвайки (2.65), имаме: $PRB_v^{work}=1-(prb^{denial})^{m_v}$ (2.66) където: $v$ е поредният номер на една от комбинациите, дефинирани чрез (2.65), като $v$ заема стойности от 1 до $2^R-1$; $m_v$ е броят на СС, в които се намира поне една от частите на комбинация с номер $v$. Тъй като участието на която и да е част в решаването на съответната функция и задача може да се разглежда като вероятност за реализиране на всички събития $R$, то можем да запишем: $PRB_{n,k}^{solv}=\sum_{v=1}^{2^R-1}(-1)^{v-1}PRB_Q^{work}$ (2.67). 2. Нееднородни СС: в този случай изразът (2.67) остава в сила, когато $PRB_Q^{work}$ ще се дава от: $PRB_Q^{work}=1-\prod_{j=1}^{L}(prb_j^{denial})^{m_{v,j}}$ (2.68), където: $m_{v,j}$ е броят на СС от $j$-ти тип, в които се намира поне една част от комбинацията $v$; $L$ – броят на различни типове СС, участващи в решението на съответната функция и задача; $prb_j^{denial}=1-prb_j^{work}$. След като заместим (2.68) в (2.67) ще получим: $PRB_{n,k}^{solv}=\sum_{v=1}^{2^R-1}(-1)^{v-1}\left[1-\prod_{j=1}^{L}(prb_j^{denial})^{m_{v,j}}\right]$ (2.69), където: $R$ е общият брой части, на които е разделена $k$-тата функция и задача; $v$ е пореден номер от $Q$; $x$ – брой на частите в комбинацията $v$. Изразът (2.69) представлява едно обобщение на (2.61), (2.64) и (2.67), които са негови частни случаи, за случаите на частично дублирани СС. За случаите на пълно дублирани СС, имайки предвид (2.39), ще имаме: $PRB_{n,k}^{solv}=\sum_{f=1}^{K+(k-1)}\sum_{(l)}\prod_{j=1}^{z}(G_{m_j})^{u_j}(prb_j^{work})^{u_j}(prb_j^{denial})^{m_j-u_j}$ (2.70), където: $f$ е номер на групата, за която е в сила следното: $EFI_{r_{l,1}}\geq\sum_{k=1}^{K+(f-1)}EFI_k^{FT}$ (2.71), имайки предвид (2.37); $r_{l,1}$ е комбинацията $l$ на състава $u_j$ при $i$-тото състояние на системата; $z$ е броят на типовете СС с $n$-то функционално название; $i$ е броят на работещите модули с $n$-тото функционално название; $u_j$ е броят на работещите СС от $i$-ти тип; $m_j$ е общият брой СС от $i$-ти тип; $prb_j^{work}$ е вероятността за работа на СС от $i$-ти тип. По такъв начин ние получихме изразите (2.59), (2.69) и (2.70), които са търсените аналитични изрази при линейна суперпозиция на ефективностите на функциите и задачите.

Нелинейна суперпозиция на ефективностите. Отчитайки корелационните зависимости между $EFI_k^{FT}$, изразът (2.59) трябва да бъде коригиран по следния начин: $EFI^{CPS}=\sum_{v=1}^{2^K-1}EFI_{F_v}^{FT}\left[\prod_{i=1}^{s}\sum_{j=1}^{v}PRB_{i,j}^{sol}-\prod_{i=1}^{s}\sum_{j=1}^{v-1}PRB_{i,j}^{solv}\right]$ (2.72), където: $F_v$ е някаква функция от $v$, зависеща от приетата таблица на приоритетите и от закона на изменение на $EFI^{FT}$; $PRB_{i,j}^{solv}$ е вероятността за решение на дадена комбинация от функции и задачи. Този израз е твърде сложен и определянето на съставната част на ефективността на системата при решаване на състав $v$ от функции и задачи става чрез: $EFI_v^{FT}=EFI_{F_v}^{FT}(PRB_v^{solv}-PRB_{v-1}^{solv})$ (2.73). Общата ефективност на системата съответно е: $EFI^{CPS}=\sum_{v=1}^{2^K-1}EFI_v^{FT}$ (2.74). Сега можем да преминем към аналитичните изрази за вероятностите, които са разделени на два основни случая: 1) Вероятност за решаване на $v$-тия състав от

задачи в i-тия СС – има две възможности: 1. $prb^{work}$=const: имайки предвид направените вече разглеждания, можем да запишем израз за случай на пълно дублирани СС: $PRB_{v,j}^{solv}=\sum_{i=1}^{U_j}\prod_{w=1}^{z}(G_{m_{j,w}})^{u_{i,j,w}}(prb_{j,w}^{work})^{u_{i,j,w}}(prb_{j,w}^{denial})^{m_{j,w}-u_{i,j,w}}$ (2.75), където: z е брой на типовете СС; $m_{j,w}$ е общият брой СС от w-тип; $u_{j,i,w}$ е брой на работещите СС от w-тип в i-тото състояние на CPS; $U_j$ е брой на състоянията на системата, осигуряващи решението на v-тия състав от задачи в даден СС от j-тип. За случай на частично дублирани СС имаме: $PRB_{v,j}^{solv}=\sum_{i=1}^{U_j}K1_i^j\prod_{w=1}^{z}(prb_{j,w}^{work})^{u_{i,j,w}}(prb_{j,w}^{denial})^{m_{j,w}-u_{i,j,w}}$ (2.76), където: $K1_i^j$ е коефициент, зависещ от разпределението на функциите и задачите по СС, като най-често има стойност единица. 2. $prb^{work}=e^{-ct}$: имайки предвид посочените по-горе разглеждания и характера на експоненциалната зависимост, имаме: $PRB_{v,j}^{solv}=\sum_{i=1}^{U_j}K2_i^j\exp\left[-t\sum_{j=1}^{z}(u_{j,i,w})(c_{j,w})\right]$ (2.77), където: $K2_i^j$ е също коефициент, зависещ от разпределението на функциите и задачите по СС и се изчислява за случай на пълно дублирани СС по следния израз: $K2_i^j=\sum_{i=1}^{U_j}\prod_{w=1}^{z}(G_{m_{j,w}})^{u_{i,j,w}}\sum_{l=0}^{m_{j,w}-u_{i,j,w}}(-1)^l(G_{m_{j,w}-u_{i,j,w}})^l(prb^{work})^{u_{i,j,w}\cdot l}$ (2.78), а в случай на частично дублирани СС - по следния израз: $K2_i^j=\sum_{i=1}^{U_j}K1_i^j\prod_{w=1}^{z}\sum_{l=0}^{m_{j,w}-u_{i,j,w}}(-1)^l(G_{m_{j,w}-u_{i,j,w}})^l(prb^{work})^{u_{i,j,w}\cdot l}$ (2.79). 2) Вероятност за решаване на v-тия състав от функции и задачи в CPS – има също две възможности: 1. $prb^{work}$=const: изразът е: $PRB_v^{solv}=\prod_{j=1}^{s}PRB_{v,j}^{solv}$ (2.80), където: s е броят на основните СС с различни функционални названия; 2. $prb^{work}=e^{-ct}$: имайки предвид разглежданията в т. 2.3.2, изразът за вероятността ще бъде: $PRB_v^{solv}=\sum_{i=1}^{M}\frac{G_i}{K3_i^2T^2}\left[1-e^{-K3_iT}(K3_iT+1)\right]$ (2.81), където: $M=\prod_{j=1}^{s}U_j$ (2.82a), $G_i=\prod_{j=1}^{s}K2_i^j$ (2.82b), $K3_i=\prod_{j=1}^{s}K4_i^j$ (2.82c), $K4_i^j=\sum_{w=1}^{z}u_{j,i,w}c_{j,w}$ (2.82d). Нека сега формулираме аналитичните условия на оптимизационния процес, т.е. критериите за постигане на оптималност в определен смисъл в структурата на CPS.

*Аналитични условия*. Съгласно [182] критериите на оптимизационен процес за относителната $EFI_{rel}^{econ}$ и абсолютната $EFI_{abs}^{econ}$ икономическа ефективност имат вида: $EFI_{rel}^{econ}=\frac{1}{T^2}\int_0^T\frac{EFI^{CPS}}{PRI^{CPS}}dt$ (2.83) $EFI_{abs}^{econ}=\frac{1}{T^2}\int_0^1 EFI^{CPS}dt-K_{pr}PRI^{CPS}$ (2.84), където: T е времето за експлоатация на CPS без прекъсване (профилактика и/или ремонт); $EFI^{CPS}$ е ефективността на CPS, изчислена по дадените по-горе изрази; $PRI^{CPS}$ е цената на CPS, изчислена след сумиране на цените на отделните СС; $K_{pr}$ е коефициент на пропорционалност, отразяващ относителната тежест на СС, и заемащ стойности от 0 до 1. За случай на линейна суперпозиция на ефективностите аналитичните условия имат вида: $EFI_{rel}^{econ}=EFI_{rel.max}^{econ}$ (2.85) $EFI_{abs}^{econ}=EFI_{abs.max}^{econ}$ (2.86). За случай на нелинейна суперпозиция на ефективностите аналитичните условия имат вида: $\sum_{v=1}^{2^k-1}\Delta EFI_v^{FT}\Delta PRB_{e,v}^{solv}\prod_{i=1}^{s}PRB_{i'e,v}^{solv}>EFI_{rel}^{econ}\Delta PRI^{CPS}$ (2.87) $\sum_{v=1}^{2^k-1}\Delta EFI_v^{FT}\Delta PRB_{e,v}^{solv}\prod_{i=1}^{s}PRB_{i'e,v}^{solv}>EFI_{abs}^{econ}\Delta PRI^{CPS}$ (2.88), където: $\Delta PRI^{CPS}$ е разликата в цените на един състав от СС преди и след изменението; $\Delta PRB_{e,v}^{solv}$ е разликата във вероятността за решение на v-тата комбинация от функции и задачи в e-тия СС, при стар и нов състав; $\Delta EFI_v^{FT}$ е разликата в ефективностите на функциите и задачите от комбинацията v при стар и нов състав. В заключение ще спомена, че изборът на критерии за оптимизационен процес е обусловен от изискването за универсалност и простота на критериите. В литературата [182, 201], съществуват описани много критерии: цена на ефективното бързодействие, надеждност, време за

решаване на задачите, тегло, обем, консумирана енергия, точност, производителност и други. Съществуват и опити за избор на интегрални критерии, в които се обединяват няколко или всички критерии за дадена структура на изследваната система, но всички те имат изкуствен характер и не спомагат за реално въздействие върху оптимизационния процес.

*Алгоритъм на оптимизационен процес за линейна суперпозиция на ефективностите.* Блоковата схема на алгоритъма има следните основни стъпки: 1) Определяне на минимален състав на CPS, който да гарантира решаването на поставените функции и задачи. Това е случай, когато не е предложен състав, чиято структура трябва да се оптимизира; 2) Определяне на матриците $CON^{CPS}$, $DIS^{CPS}$, броя на основните СС (s), таблиците на характеристиките на отделните функции и задачи и на характеристиките на отделните СС ($TAB^{FT}$ и $TAB^{CC}$), коефициента на отношение на сумата от цените на всички основни СС към цената на най-скъпия СС ($K_{val}$) и цената на неосновните СС ($PRI^{CC}_{anbas}$); 3) Изчисляване на $EFI^{CPS}$ (2.59), $EFI^{econ}_{rel}$ (2.83) и $EFI^{econ}_{abs}$ (2.84), като величините се определят в зависимост от j (номера на типа СС) и $m_j$ (броя на СС от даден тип); 4) Сравнение на критерия за оптимизационен процес $COP^{CPS}_{max}$ с $COP^{CPS}_{i}$ и съответно разклоняване към стъпка 5 или 6; 5) Запис в паметта на текущата стойност на COP като максимална, спрямо която ще се сравняват вариантите; 6) Сравнение на $EFI^{CPS}_{i}$ с $EFI^{CPS}_{max}$ спрямо зададена $\Delta EFI$ и съответно разклоняване към стъпка 3 или 7; 7) Сравнение на текущия номер на типа на СС (j) с общия брой разполагаеми типове СС (z) и съответно разклоняване към стъпка 8 (ако е по-малко или равно) и към стъпка 9 (ако е по-голямо); 8) Подготовка за избор на нов тип СС (преадресиране) и връщане към стъпка 3; 9) Избиране на максимална стойност на COP от частните максимуми за всеки тип СС; 10) Определяне състава на СС и изчисляване на съответните $PRB^{sch}_{s}$; 11) Определяне на оптимална система от СС по отношение на зададена сигурност. Това е подпрограма, чийто алгоритъм представлява начално приближение на търсеното решение за оптимална структура по отношение на $EFI^{econ}_{rel}$ и $EFI^{econ}_{abs}$; 12) Проверяване на възможните изменения на текущия състав за всички основни СС, т.е. търсене на разликата между номера на проверения СС (i) и броя на основните СС (s) и съответно разклоняване към стъпка 18 (ако е по-голяма) или стъпка 13 (ако е по-малка или равна на нула); 13) Проверяване на всички възможни комбинации за изменение състава на системата по i-тия СС. За тази цел номерът на комбинацията j се сравнява с израза z(v+1), където z е брой на типовете на i-тия СС, а v е броят на типовете в дадена структура. Ако j е по-голямо от израза, следва преход към стъпка 17, в противен случай следва стъпка 14; 14) Изменение на текущия състав на i-тия СС и преизчисление на стойността на COP; 15) Сравнение на новата получена стойност $COP^{CPS}_{i,j}$ с фиксираната в паметта $COP^{CPS}_{max}$ Ако новата стойност е по-малка от максималната, следва преход към стъпка 13, в противен случай следва стъпка 16; 16) Записване на състава на i-тия СС и замяна на максималната стойност с $COP^{CPS}_{i,j}$, след което следва преход към стъпка 13; 17) Фиксиране на оптималния състав на i-тия СС, преадресиране и подготовка на програмата за следващия СС, след което следва преход към стъпка 12; 18) Окончателно формиране на системата като текущо приближение и преход към стъпка 19; 19) Сравнение на текущото приближение за състава на системата с предходното. Ако съставите са еднакви, следва стъпка 21, ако са различни, следва стъпка 20; 20) Подготовка на програмата за преход към стъпка 12, т.е. търсене на ново приближение на състава; 21) Извеждане на окончателния резултат за структурата на CPS и край на стъпките.

*Алгоритъм на оптимизационен процес за нелинейна суперпозиция на ефективностите.* Блоковата схема на алгоритъма има следните основни стъпки: 1) Проверяване на входните данни, задаващи системата, чиято структура ще се оптимизира, като тези данни включват: 1. Общи характеристики на системата: общ брой на СС в системата, брой на функционалните названия на СС, брой на основните и неосновните СС, брой на еднородните и нееднородните СС, брой и характер на дублираните СС; 2. Частни характеристики на системата: номер на функционалното название на СС, номер на типа на СС, брой на СС от даден тип, признаци за основен и неосновен СС, за пълно и частично дублирани СС; 3. Общи характеристики на функциите и задачите: общ брой на функциите и задачите, брой на типовете функции и задачи, брой на основните и неосновните функции и задачи; 4. Частни характеристики на функциите и задачите: бързодействие и разрядност на процесорите, бързодействие и обем на видовете памети, скорост и начин на предаване на информацията от комуникационните входно/изходни

устройства, вид и характер на периферните входно/изходни устройства, вид и характер на диалоговите устройства, вид и характер на превключвателите, маршрутизаторите, концентраторите, шинните системи за връзка и други; 5. Общи характеристики на СС: за всеки тип СС се задават неговата цена, обем, тегло и надеждност (като $prb^{work}$ – вероятност за безотказна работа за време t или като c – интензивност на отказите; 6. Частни характеристики на СС: за всеки тип СС се задават неговата основна функционална характеристика (например за процесорите – бързодействие, умножено по разрядност), както и други характеристики, имащи отношение към основната характеристика (например брой и разрядност на вътрешни регистри); 7. Общи характеристики на оптимизационния процес: закон на изменение на ефективностите на отделните функции и задачи, вид на избрания принцип на приоритетност, като за отделните задачи може да съществува повече от един принцип; 8. Частни характеристики на оптимизационния процес: вид на критериите на оптимизационния процес, техните формули за изчисление и съпътстващи величини (за $COP_{rel}^{econ}$ и $COP_{abs}^{econ}$, това са T - времето на експлоатация без профилактика и ремонт и $K_{pr}$ - коефициент на пропорционалност, показващ относителния характер на цените на СС). Ако всички входни данни са налице, т.е. системата е зададена, следва преход към стъпка 10, в противен случай следва стъпка 2. 2) Присвояване на номера на всички функционални названия на СС, като в зависимост от зададените основни СС се определят техните съответни номера (от 1 до 10). Ако текущият номер е по-малък или равен на 10, следва преход към стъпка 3, в противен случай следва преход към стъпка 6; 3) Определяне на съответна характеристика на функцията и задачата за текущия номер на функционалното название на СС и, ако характеристиката е основна, следва преход към стъпка 4, в противен случай следва стъпка 5; 4) Определяне на типовете минимални количества СС с дадено функционално название, чийто сумарни характеристики осигуряват изпълнението на функцията и задачата. Полученият резултат се записва в таблицата с характеристиките на системата, като заедно с това се контролира броят на основните ($M_1$) и броят на частично дублираните ($M_2$) СС; 5) Определяне на СС при изискването за минимална цена и съответно запомняне на състава в паметта; 6) Допълване на таблицата с характеристиките на системата и с характеристиките на неосновните модули. Тази стъпка се реализира след анализа на първите десет функционални названия; 7) Определяне на необходимостта от използване на превключвател SWI в системата, ако е налице повече от една шинна система. Ако е необходим превключвател, следва стъпка 8, в противен случай следва стъпка 9; 8) Определяне на типа и състава на превключвателя, както и таблицата на превключванията във времето; 9) Определяне на типа, характера и комплектацията на пулта за управление (ако е предвиден), както и на една общосистемна характеристика - основната функционална принадлежност на CPS; 10) Определяне на последователност от номера на съчетания от функции и задачи въз основа на закона на изменение на ефективностите и приетия принцип на приоритетност, като се осигурява записване на резултатите в паметта; 11) Проверяване на номера на основния частично дублиран СС - ако е по-голям от 5, следва стъпка 13, в противен случай следва стъпка 12; 12) Разпределение на обема на паметта по отделните функции и задачи въз основа на следното: 1. Минимален брой СС за една функция и задача. Най-добре ще бъде, ако всяка функция и задача е изцяло разположена в един СС, без да се налага да бъде разделяна на части; 2. В един СС се разполага само една функция и задача (или част от един номер); 3. Функции и задачи с висок приоритет се разполагат по възможност в СС с висока надеждност; 4. Максимално използване на ресурсите на СС. 13) Последователно изменение с единица на номера на съчетанието, (състава) от задачи от 1 до $2^K$-1, където K е общият брой задачи. Конкретният номер v съответствува на конкретното съчетание от задачи, записано в паметта при стъпка 10. Ако v е по-малко от $2^K$, следва преход към стъпка 14, в противен случай следва стъпка 18; 14) Последователно изменение с единица на j - броя на разглежданите СС от 1 до $M_1$. Ако j е по-малко или равно на $M_1$, то следва преход към стъпка 15, в противен случай следва преход към стъпка 16; 15) Определяне на вероятността за решаване на v-тия състав от задачи в j-тия СС по формули от (2.75) до (2.79), след което - стъпка 14; 16) Определяне на вероятността за решаване на v-тия състав от функции и задачи в системата по формули от (2.80) до (2.82); 17) Определяне на съставната част от ефективността на системата при v-тия състав от функции и задачи по формула (2.73); 18) Определяне на общата ефективност на системата по формула (2.74); 19) Определяне на критерия на оптимизационния процес по формули (2.83) или (2.84) или по някакви други функционални зависимости в други случаи; 20) Проверяване на номера на основния СС. Ако j е по-

малко или равно на $M_1$, то следва преход към стъпка 21, в противен случай следва стъпка 26; 21) Определяне на броя на възможните варианти на изменение на състава на j-тия СС по следната формула: $Z_j=z_j(z_j+1)$ (2.89), където $Z_j$ е изменението на ефективностите на задачите за j-тия тип СС, $z_j$ е броят на различните типове от j-тия СС; 22) Проверяване на номера на варианта на изменение на състава на СС чрез преброяване на направените проверки на изменения състав на j-тия СС. Ако този брой е по-малък или равен на $Z_j$, следва преход към стъпка 23, в противен случай следва стъпка 24; 23) Проверяване на условието на нарастване на критерия за оптимизация чрез (2.87) или (2.88). Ако проверката е положителна, се прави запис на новия състав на j-тия СС и на новата стойност на лявата страна на неравенствата, след което следва връщане към стъпка 22; 24) Определяне на максималното нарастване на COP чрез сравнение на записаните в паметта леви страни на неравенствата (2.87) и (2.88); 25) Записване на изменения състав на j-тия СС; 26) Проверяване за наличие на изменение в състава на системата. Ако има изменение, следва преход към стъпка 27, а ако не, следва стъпка 28; 27) Проверяване за наличие на изменение в частично дублираните СС. Ако има изменение - преход към стъпка 11, а ако не - стъпка 13; 28) Фиксиране и извеждане на крайните резултати от оптимизационния процес за структурата на CPS: състава на системата, разпределението на функциите и задачите в частично дублираните СС, общата ефективност на системата, стойностите на COP, процента на изменение на ефективността на системата по отношение на началния състав, процента на увеличение на COP по отношение на началния състав, таблицата на превключване, ако в системата е въведен превключвател, и всички начални данни.

МЕТОДИКА НА ОПТИМИЗАЦИОНЕН ПРОЦЕС ЗА УПРАВЛЕНИЕТО. Въпросът за принципите на управление на CPS е от особено значение за нейното оптимално проектиране. Това е така, защото, ако разгледаме нещата в исторически аспект, то антивирусните програми са били създадени като подчинена част на операционната система. С развитието на информационните технологии стана възможно те рязко да намалят зависимостта си от операционната система и заедно с това да разширят многократно контролираните зони и сектори и изпълняваните функции и задачи. По такъв начин естествено се извърши и преходът от отделна защитна програма към система от защитни програми, а оттам - до създаване на CPS, които включват в себе си не само софтуерни ресурси, но и значително количество хардуерни ресурси. Всичко това означава, че така образувалата се автономна система, влизаща в състава на защитаваната система, има остра необходимост от правилно избрано управление. Основните възможности включват избор и определяне на правилни управляващи съотношения с мрежовата операционна система, локалната операционна система и периферната операционна система. По-нататък решаващо значение има необходимата степен на автономност в различните етапи от работата на CPS. Явно е, че различните потребители имат различна готовност да платят цената, която се изисква за различните варианти. Балансът между изчислителните и обработващи функции, задачи, процедури, операции и други изисква решение за всеки конкретен случай при проектирането на CPS. Това означава използване и прилагане на различни специални методи. Един от тях, който ще използваме, е системният подход, при който проблемите по оптимизацията на състава, структурата и управлението на CPS се решават чрез прилагане на подоптимизации и на комплексен анализ на взаимоотношенията между обектите на оптимизация. В този смисъл подреждането не бива да се разглежда като задължително и последователността в прилагането на трите методики на оптимизационен процес е в зависимост от конкретните задачи, като се допуска и самостоятелно приложение. Основната задача, която се поставя пред CPS в процеса на нейната работа е: да се изгради релацията **задача - решение** при изисквания за минимално време и минимални ресурси. Първото изискване означава времето за реакция на системата да бъде достатъчно малко в сравнение с времето за реакция на оператора. Второто изискване означава минимум управляваща информация за дадено количество приложна информация (касаеща пряко процеса). По такъв начин, задачата за оптимизационен процес на управление се свежда до задачата за оптимизиране на съотношението управляваща/приложна информация при минимално време за реакция на CPS. Най-подходящите за нашите цели методи за решаване на тази задача, съгласно направения обзор в глава първа, са числените итерационни методи.

*Аналитични изрази.* Нека да е зададен някакъв процес, който включва в себе си множество от условия (например, присъствие, отсъствие или поява на определени шестнадесетични или двоични последователности и/или определена "агресивна" поредица от действия) и тяхното изпълнение за минимално време с минимални средства. По същество това е оптимизационна задача на динамичен

процес, която може да се представи чрез целева функция и в съответствие с [55]: $TFOP=BF_0^1[vs(t_m),t_m]+\int_{t_s}^{t_m} BF_0^2[vs(t),vm(t),t]dt$ (2.90). От теорията на вариационните изчисления и задачите за оптимално управление е известно, че (2.90) може да съществува в три разновидности. По такъв начин, задачата за оптимизация на управлението на един динамичен процес може да съществува в три разновидности, известни като: 1) Задача на Болц, при нея $BF_0^1 \neq 0$ и $BF_0^2 \neq 0$; 2) Задача на Лагранж, при нея $BF_0^1=0$; 3) Задача на Майер, при нея $BF_0^2=0$. Интерес за нас представлява задачата на Майер, при която (2.90) има вида: $TFOP=BF_0^1[vs(t_m),t_m]$ (2.91). Нека процесът да се представя с една система от линейни векторни диференциални уравнения, която описва съвкупност от състояния по следния начин, при което (') ще означава производна по времето (t): $VS'=F[vs(t),vm(t),t]$ (2.92) със следните гранични условия: $v_{start}m_i[vs(t_0),t_0]=0$, $i=1,...,p \le a$ (2.93) $v_{end}m_j[vs(t_m),t_m]=0$, $j=1,...,r \le a$ (2.94). При тази постановка ще търсим такъв вектор на управление, който да осигурява екстремум на целевата функция при следния критерий на оптимизационен процес: $COP_{min}=BF_0^1[vs(t_m),t_m]$ (2.95). Този COP има смисъл само тогава, когато функцията $vs(t_m)$ е свободна, т.е. $vs(t_0)=vs_0$ (2.96) като се предполага, че времето $t_m$ е зададено. Освен това, предполага се, че векторът на управление е записан във вида: $vm(t)=[vm_1(t),...,vm_b(t)]^T$ (2.97) и принадлежи към някаква област, зададена по следния начин $cm_j^{min} \le vm(t) \le cm_j^{max} \rightarrow j=1,2,...,b$ (2.98). По такъв начин, проблемът за оптимизация може да се дефинира така: да се определи такъв вектор на управление $vm(t)$, при който системата от (2.92), от зададено начално състояние (2.96), за зададено време $t_m$, ще бъде изведена в крайно състояние $vs(t_m)$, като се осигури минимум на COP. Това е математическият смисъл на задачата за оптимизация. Нейният физически смисъл е следният. Както вече беше посочено, в CPS трябва да се изгради съответствие между дадени условия (задачи) и тяхното изпълнение (решения), при което условията могат да се изменят във всеки следващ момент от времето. Това са всъщност елементите на вектора на състоянията. Тъй като съотношението между управляващата и приложната информация трябва да е оптимално за всеки момент от време, то управляващата информация представлява всъщност търсения вектор на управлението. Двата вектора са съответно a-мерен и b-мерен, при което за измерителна единица ще се използва "байт" или нейните кратни.

*Аналитични условия.* Аналитичното решаване на така формулираната задача за оптимизация е трудно и възможно само в някои случаи. Затова се използват числени итерационни методи, като за целите на нашето изследване е най-добре да се приложи методът на градиента. При него в качеството на начално приближение се избира една стойност на вектора на управление (или както още се нарича, функция на управлението), която ще означим като $vm_0(t)$ и за нея се изчисляват последователни корекции по такъв начин, че COP да придобива минимална стойност в посока на своето най-голямо изменение, т.е. в посока на своя градиент. Нека $vm_0(t)$ да е известна в качеството на начално приближение. Следователно може да се изчисли съответният вектор на състоянията $vs_0(t)$ и съответният $COP_0$. Тъй като, естествено, стойността на критерия не е минимална, то е необходимо да се изчисли съответната корекция към вектора на управление $\Delta vm_0(t)$, но по такъв начин, че COP да измени своята стойност в посока на своя минимум, т.е. на всяка стъпка на итерационния процес трябва да е в сила условието: $COP^{k-1}-COP^k<0 \rightarrow k=0,1,2...$ (2.99), където $COP^k=COP(vm^k)$ (2.100) представлява намерената стойност на COP. За да се изчисли следващата стойност на COP, трябва да се намери корекцията на вектора на управление, т.е. $COP^{k-1}=COP(vm^{k-1})$ (2.101), където $vm^{k-1}=vm^k+\Delta vm^k$ (2.102). Ако разложим (2.95) в ред на Тейлор по отношение на $vs^k(t_m)$ и се ограничим само с линейната част, ще получим: $COP^{k-1}=COP^k+\left[\frac{COP}{vs^k(t_m)}\right]^{t_m}\Delta vs^k(t_m) \rightarrow k=0,1,2,...$ (2.103) т.е. $\Delta COP^k=\left[\frac{COP}{vs^k(t_m)}\right]^{t_m}\Delta vs^k(t_m)$ (2.104). За да можем да изследваме $\Delta COP^k$, ще въведем една линеаризирана по отношение на $vs^k(t)$ разновидност на (2.92), която има следния вид: $\Delta VS'^k=A^k(t)\Delta vs^k+B^k(t)\Delta vm^k$ (2.105), където $A^k(t)=\frac{F}{vs^k}$ (2.106), $B^k(t)=\frac{F}{vm^k}$

(2.107). Тъй като (2.96) е в сила и за (2.105), то $\Delta vs^k(t_0)=0$ (2.108). При тези условия, решението на (2.105) има вида: $\Delta vs^k(t)=\int_{t_0}^{t_m} MAT^*_{fund}(t_m,t)B^k(t)\Delta vm^k(t)dt$ (2.109), където $MAT^*_{fund}$ е фундаментална матрица на (2.105). Ако в (2.104) заместим с гореполучения израз (2.109) и като се вземе предвид (2.95), след съответните преобразувания ще получим: $\Delta COP^k=\int_{t_0}^{t_m}\left[\left[MAT^k_{fund}(t_m,t)\right]^{t_m}\frac{BF^l_0}{vs^k(t_m)}\right]B^k(t)\Delta vm^k(t)dt$ (2.110). Ако се въведе една спрегната на (2.105) система, то нейното решение ще има следния вид, при което (+) ще означава спрегнатост: $VS^{-k}(t)=\left[MAT^*_{fund}(t_m,t)\right]^{t_m}VS^{-k}(t_m)$ (2.111), при което крайното състояние $VS^{-k}(t_m)$ ще се определя от $VS^{-k}(t_m)=-\frac{BF^l_0}{vs^k(t_m)}$ (2.112). Нека сега да въведем функцията на Хамилтън [33] $H(vs,vm,vs^-,t)=VS^{-t}*F(vs,vm,t)$ (2.113). Ако се вземе предвид (2.111), (2.112), и (2.113), то за (2.110) може вече да се запише: $COP^k=-\int_{t_0}^{t_m}\left[\frac{H}{vm^k(t)}\right]^{t_m}\Delta vm^k(t)dt$ (2.114). По такъв начин, постигнахме поставената задача, а именно, корекцията на минимизирания COP е изразена като функция на корекцията на вектора на управлението. С цел да се ограничи нарастването на корекцията на вектора за управление се въвежда следното ограничаващо условие $\int_{t_0}^{t_m}\left[\Delta vm^k(t)\right]^{t_m}MAT_{inf}\Delta vm^k(t)dt=v^2\rightarrow k=0,1,2,...$ (2.115), където $v^2$ е зададена константа, а $MAT_{inf}$ е постоянна матрица на влиянието, наречена така, защото придава различно тегло на отделните координати на управление. като за повечето случаи се избира тя да бъде единична. Решението на задачата за минимизиране на (2.114) при спазване на условието (2.115) и в съответствие с направените извеждания в [44] за L има вида: $\Delta vm^k(t)=-\frac{1}{2L}MAT^{-1}_{inf}\frac{H}{vm^k(t)}$ (2.116), където $L=\pm\frac{1}{2|v|}\sqrt{\int_{t_0}^{t_m}\left[\frac{H}{vm^k(t)}\right]^{t_m}MAT^{-1}_{inf}\frac{H}{vm^k(t)}dt}$ (2.117), при което знакът на (2.117) се избира да бъде отрицателен, ако трябва корекцията на *COP* да е минимална. Ако се обединят постоянните множители в (2.116) и (2.117) в една нова константа, която можем да наречем дължина на стъпката d, то за корекцията на $vm^k(t)$ и за търсената минимална корекция на $COP^k$ ще получим $\Delta vm^k(t)=d^kMAT^{-1}_{inf}\frac{H}{vm^k(t)}$ (2.118) $\Delta COP^k=-d^k\int_{t_0}^{t_m}\left[\frac{H}{vm^k(t)}\right]MAT^{-1}_{inf}\frac{H}{vm^k(t)}dt$ (2.119). Константата d може да бъде положителна или отрицателна. Изборът й се определя от това, дали се търси максимум или минимум на функцията на Хамилтън. За нашите цели, константата d е положителна. В резултат на горните разсъждения е възможно да се определи последователност от действия на методика за оптимизационен процес, чрез които да се реши задачата за оптимизация на вектора на управлението при зададена точност чрез един итерационен процес.

*Алгоритъм на оптимизационен процес*. Основни стъпки са: 1) Избиране на начално приближение за вектора (функцията) на управлението; 2) Интегриране на системата (2.92) в права посока (от $t_0$ към $t_m$) при спазване на началното условие (2.96) чрез избран метод за числено интегриране със съответен брой итерации; 3) Интегриране на спрегнатата на (2.92) система в обратна посока (от $t_m$ към $t_0$), като се има предвид (2.112); 4) Изчисляване на градиента на функцията на Хамилтън; 5) Определяне на корекцията на вектора на управлението при избрани дължина на стъпката и матрица на влиянията; 6) Повтаряне, започвайки от стъпка 2, на изчисляването на функцията на управлението, докато се достигне зададената точност. Програмната реализация има една основна програма и няколко подпрограми за числено интегриране (при съответно избран метод за числено интегриране), със средствата на PASCAL/DELPHI. Основната програма има три цикъла: 1) Вътрешен, с индекс c, който реализира оптимизацията на дължината на стъпката на итерационния процес; 2) Среден, с индекс k, който реализира изчислението на корекцията на вектора на управлението; 3) Външен. с индекс h, който реализира многократно проиграване на алгоритъма на методиката за различни примери и за различни входни данни. Като метод

за числено интегриране е избран методът на Рунге-Кут [33], който отговаря на конкретните изисквания в нашия случай и освен това влиза в състава на библиотеките от стандартни програми за много компютърни системи. Програмната реализация започва със задаване на: 1) a – броя на диференциалните уравнения, които ще трябва да бъдат интегрирани; 2) b – размерността на вектора на управление; 3) n – брой на стъпките на интегриране. След това се продължава с резервиране на полета в паметта със съответната размерност: 1) (a,n+1) – за вектора на състоянието; 2) (a,n+1) – за спрегнатия вектор на състоянието; 3) (b,n+1) – за вектора на управлението; 4) (b,n+1) – за корекцията на вектора на управлението; 5) (a,1) – за вектора на крайните условия; 6) (a,1) – за вектора на началните условия; 7) (b,b) – за матрицата на влиянието, ако тя не е единична. Следващото действие е въвеждането на входните данни: *1)* Параметри на началните условия; 2) Параметри на крайните условия; 3) Коефициентите на диференциалните уравнения; 4) Константите за числено интегриране; 5) Брой на примерите, които ще се решават ( $h_{max}$ ); 6) Брой на итерациите, необходими за всеки пример ( $k_{max}$ ); 7) Условието за необходимата точност на итерацията ( v ); 8) Условието за прекъсване на итерациите при несходимост на процеса ( w ). Изчислителният процес започва с изчисление на началното приближение за вектора на управление. Едновременно с първото интегриране на диференциалните уравнения на основния и на спрегнатия вектор на състоянието, започва да работи програмата за итерациите. Средният цикъл продължава, докато не се постигне зададената точност или зададения брой итерации. При несходимост итерацията се прекъсва. В точките, в които приключва своето действие средният програмен цикъл, се решава, дали е завършено изпълнението на програмата за даден пример или ще се започне нов изчислителен процес за нов пример.

## ГЛАВА ТРЕТА

ОСНОВНА КОНЦЕПЦИЯ ЗА МОДЕЛ. Като изходни предпоставки за създаване на основната концепция за модел на CPS ще се използват някои положения от теорията и практиката на микропроцесорните и микрокомпютърните изчислителни системи [1, 111, 112, 130, 132, 150, 154, 156, 186, 215, 249, 251], на системите за телекомуникация [15, 30, 52, 77, 85, 87, 88, 158, 159, 199, 252, 267], на системите за антивирусна, компютърна и мрежова защита [2, 3, 4, 31, 57, 58, 59, 60, 104, 105, 119, 138, 160, 193, 256], както и постиженията в областта на идентификацията [219, 221, 222, 224, 227, 228, 233, 247, 248, 274], ауторизацията [126, 129, 139, 157, 172, 173, 176, 207, 250, 275] и криптографията [202, 216, 238, 239, 258, 259, 291]. CPS, разглеждана като обект на изследване, може да бъде представена със следното множество от десет компонента: FT, PO, SG, NC, PC, HT, ST, DT, PH, PS, където: FT са функциите и задачите, възложени за изпълнение на CPS; PO са процесите и операциите, на които се разлагат функциите и задачите с цел тяхното изпълнение; SG са сегментите и групите (елементи на информационните потоци) и представляват групирания на информацията по определени правила; NC са мрежовите управления, които реализират координацията и придвижването на информацията на мрежово (Internet/Extranet/Intranet) ниво; PC са протоколните управления, включващи конкретните правила за транспортиране и преобразуване на информационните потоци; HT са хардуерните средства, включващи локалните системи и устройства от интегрален, дискретен, активен и пасивен тип; ST са софтуерните средства, включващи локалните системи и програми от операционен, приложен, библиотечен и драйверен тип; DT са диалоговите средства, включващи системи и устройства от екранен, гласов, картов и биометричен тип; PH е периферният хардуер включващ, периферни системи и устройства от входен, изходен, входно-изходен и преобразуващ тип; PS е периферният софтуер, включващ периферни системи и програми от операционен, приложен, библиотечен и драйверен тип. Симулационният модел, който трябва да се създаде за изследване на CPS, трябва да отразява както характерните особености на десетте съставни елемента поотделно, така и на техните взаимоотношения като цяло. За тази цел, чрез детайлно проучване на характеристиките на съставните елементи, трябва да се реализират техните формални модели, които чрез взаимодействието помежду си да позволяват създаването на един общ модел на CPS. Концепцията може да се опише формално със следните означения: с $MOD^{CPS}$ е означен общият модел на CPS; с $M^{FT}$ – моделът на функциите и задачите; с $M^{PO}$ – моделът на процесите и операциите; с $M^{SG}$ – моделът на сегментите и групите; с $M^{CPS}$ - моделът на CPS, обединяващ следващите седем модела; с $M^{NC}$ – моделът на мрежовите управления; с $M^{PC}$ – моделът на протоколните управления; с $M^{HT}$ – моделът на хардуерните средства; с $M^{ST}$ - моделът на софтуерните

средства; с $M^{DT}$ - моделът на диалоговите средства; с $M^{PH}$ - моделът на периферния хардуер; с $M^{PS}$ - моделът на периферния софтуер. В общия модел на CPS, $M^{FT}$ ще моделира работното натоварване на цялата система, като неговите входни данни са активни $IAD_a^{FT}$ и пасивни $IPD_p^{FT}$, а изходните резултати са финални $OFD_f^{FT}$ и междинни $OMD_m^{FT}$. По подобен начин са означени входните данни и изходните резултати и на останалите съставни елементи. С малките букви a, p, f и m, разположени като долни индекси, са означени съответните текущи стойности. От взаимоотношението на $M^{FT}$, $M^{PO}$, $M^{SG}$ и $M^{CPS}$ се получава общият модел на системата, означен с $MOD^{CPS}$, за който е в сила равенството $F = f(IAD_a, IPD_p, OFD_f, OMD_m)$, представляващо математичен запис на процесите.

МЕТОДИКА ЗА СИМУЛАЦИОННО МОДЕЛИРАНЕ. Методиката, която се предлага в настоящата работа, ще се гради върху основната концепция за модел на CPS. Заедно с това при нейното създаване са използвани и някои от постиженията и идеите в [24, 25, 27, 34, 36, 39, 55, 56, 65, 66, 76, 100, 108, 109, 114, 115, 117, 118], като се е подхождало съобразно целите и задачите на дисертацията. Съгласно методиката процесът на симулационно моделиране на CPS съдържа пет етапа:

*Подготовка*: 1) Дефиниране на проблема. Тук се включва общото запознаване с изследвания обект, установяване на неговите известни и неизвестни характеристики и параметри, техните известни математически взаимоотношения, формулиране на основните цели, както и на ефекта от разрешаването на проблемната ситуация; 2) Избор на метод за изследване. При тази стъпка се избира методът на симулационното моделиране, преценява се дали има смисъл за някои от изследванията да се използват аналитични методи, като се направи сравнение между резултатите; 3) Прогноза за възможните решения. Очертава се кръгът от възможните решения на проблема при наличието на съответна база за изследвания и експерименти; 4) Избор на база за изследване. В зависимост от актуалността на проблема и наличните възможности се определя типа и конфигурацията на моделиращата изчислителна система; 5) Определяне на необходимите ресурси. При така направените предположения и известни изчислителни възможности се определят необходимите парични средства и компютърни ресурси.

*Формиране на съставни модели.* За всеки от съставни модели, са предвидени по шест стъпки, които ще разгледаме накратко за един от тях, например за $M^{HT}$: 1) Създаване на концептуален модел. Като се има предвид състава на хардуерните средства и тяхната конфигурация на свързване, се изгражда концепция за симулационен модел при спазване изискванията на симулационното изследване; 2) Определяне на критериите за оценка на модела. При тази стъпка се определят онези критерии за оценка на $M^{HT}$, които имат най-пряко отношение към изследвания проблем и позволяват най-реална връзка с обекта; 3) Определяне на съотношенията между критериите за оценка. Тъй като много често факторите, касаещи проблема, имат динамичен характер, то се налага отделните критерии, имащи понякога статичен характер, да се свържат в определени съотношения, съответстващи на динамиката на процесите в обекта; 4) Съставяне на моделиращ алгоритъм. Въз основа на описаните по-горе стъпки се прави алгоритъм, моделиращ функционирането на хардуерните средства със съответното приближение; 5) Реализиране на работен модел. Това е описанието на алгоритъма от горната стъпка със средствата на съответно избран език по начин, който позволява програмна реализация върху моделиращата система; 6) Проверка за достоверност на модела. След поредица от проигравания на програмната реализация на $M^{HT}$ се прави анализ на резултатите, като се сравняват с поведението на реалния обект или подобен на него със съответните допускания.

*Експерименти със съставни модели.* Този етап се състои от пет стъпки, които се прилагат за всеки един от съставните модели. Ще разгледаме стъпките, като за пример отново ще бъде използван $M^{HT}$: 1) Планиране на експериментите. При тази стъпка съобразно поставените задачи (касаещи $M^{HT}$ и $M^{CPS}$) се определя характерът и броят на експериментите; 2) Задаване входните данни на параметрите на модела. При наличните възможности на моделиращата система за въвеждане на данни и при установени диапазони на изменения на параметрите се задават конкретни числени стойности; 3) Проиграване на модела и събиране на резултатите. В съответствие с направените досега разглеждания и при осигурено компютърно време за моделиращата изчислителна система се прави проиграване на модела определен брой пъти при съответен пакет входни данни, като се осигуряват условия за безконфликтно събиране на изходните резултати; 4) Анализ на резултатите. Като се имат предвид характера и количеството на резултатите, е възможно при голям брой резултати да се приложат методи за автоматизирана статистическа обработка или, ако опитът и интуицията на проектанта позволяват, анализът може да се

извърши по чисто емпиричен път; 5) Оценки и изводи. Тук се прави оценка, дали поставените задачи са решени чрез проведените експерименти и, ако е така, какви изводи могат да се направят конкретно за $M^{HT}$ или за CPS. Ако задачите не са решени, се определя какво и къде трябва да се промени.

*Формиране на общия модел.* Съобразно възприетата основна концепция, общият модел ще бъде съставен от десет части и на този етап, ще се извърши тяхното обединяване. При този процес ще се търси разкриване на тяхното взаимодействие в динамичен режим, при което чрез отпадане на някои подробности и параметри на съставните модели ще се акцентира върху общите за CPS логически, функционални, времеви и структурни особености в съответствие с поставените цели и задачи на симулационното изследване. Тъй като шестте стъпки на този етап бяха разгледани вече, няма да се спираме върху тях, като ще отбележим само, че те не представляват механичен сбор от съответните стъпки на съставните модели и в процеса на формиране на общия модел може да се наложи да бъдат изменени много неща от отделните модели, като по такъв начин се осигурява една обратна връзка във веригата на симулационното изследване.

*Експерименти с общия модел.* Позовавайки се на основната концепция и на разглежданията по-горе и, разбира се, в съответствие с поставените общи цели и задачи на симулационното изследване, този етап ще се състои от познатите пет стъпки, подобно на всички предишни. По отношение на анализа на резултатите ще се търси изпълнение на поставените основни цели, като получените решения по симулационен път могат да бъдат сравнени при възможност и целесъобразност с аналогични или приблизителни решения, получени по аналитичен път, или с преки измервания върху аналогичен или подобен реален обект.

ОПТИМИЗАЦИЯ ПО ОПРЕДЕЛЕНИ КРИТЕРИИ. Оптимизационните процеси могат да се разглеждат като нещо неразривно свързано с почти всички процеси на изследване и проектиране на реални обекти. Това особено силно важи за микроизчислителни системи, каквито са всъщност съвременните CPS, защото благодарение на модерната технология проектантът има на разположение много голям брой компоненти, които могат да се обединят в различен състав, количество и конфигурации. Естествен е изводът, че в тази огромна съвкупност от варианти само методите и средствата на оптимизацията по определени критерии могат да помогнат за намиране на оптималното (в определен смисъл) решение на поставената задача.

*Математическо описание на обекта на оптимизация.* Тъй като ще използваме понятията параметри и характеристики в по-нататъшните разглеждания, позовавайки се на [125, 136, 142, 143, 155, 162, 187, 198, 200, 208, 220, 229, 231, 237, 240], ще направим следното разграничение между тях. Всички свойства на разглеждания обект, които могат да бъдат разглеждани като първични, ще се наричат параметри, а всички останали свойства, които могат да се разглеждат като вторични (функции на параметрите) ще се наричат характеристики. Както беше посочено вече, CPS може да се опише с десет съставни елемента като всеки от тях може да се разглежда като обединение на няколко типа множества от различни типове характерни компоненти. В съответствие с тази постановка ще приемем, че изменението на типовете множества ще става по индекс $i$, а изменението на типовете характерни компоненти ще става по индекс $j$, при което с помощта на цифри, поставени като горен индекс, ще означим принадлежност към конкретен тип множество или тип характерен компонент. Изменението ще става в някаква област, която ще означим съответно с индекс $I$ и индекс $J$, като цифрата след него отново ще означава принадлежност към конкретен тип множество или тип характерен компонент. Ще се уговорим също така, да използваме като означения абревиатури от наименованията на множествата и характерните компоненти, при което абревиатура с малки букви ще означава тип множества или тип характерни компоненти, абревиатура с главни букви ще означава съвкупност от еднотипни множества или еднотипни характерни компоненти, а със символа $\in$ ще означим принадлежност към някаква област на групирани стойности.

Описание на съставен елемент "Функции и Задачи" (FT). Това е обединение на множества от различни типове функции и задачи, като в него се съдържат в неявен вид общите правила за преобразуване на входната информация до изходна, което определя най-общо предназначението на CPS. В съответствие с горното можем да запишем:

$$FT = [INF_i^1, OUF_i^2, INT_i^3, OUT_i^4]_{i(1\div4)}^{\in J(1\div4)} \quad (3.2a) \qquad FT = f(INF, OUF, INT, OUT) \quad (3.2b)$$

$$INF = (\mathrm{inf}_j^1)_{j1}^{\in J1} \quad (3.2c) \quad OUF = (ouf_j^2)_{j2}^{\in J2} \quad (3.2d) \quad INT = (\mathrm{int}_j^3)_{j3}^{\in J3} \quad (3.2e) \quad OUT = (out_j^4)_{j4}^{\in J4} \quad (3.2f)$$

Описание на съставен елемент "Процеси и Операции" (PO). Това е обединение на множества от различни типове процеси и операции, като в него се съдържат разложени на съответни съставящи

планираните за реализиране функции и задачи. Процесите и операциите са едно преходно състояние на функциите и задачите, което подпомага прехода им към следващия съставен елемент под формата на сегменти и групи от информационни последователности. В съответствие с горното можем да запишем:

$$PO = [INP_i^5, OUP_i^6, INO_i^7, OUO_i^8]_{i(5\div8)}^{\in I(5\div8)} \quad (3.3a) \qquad PO = f(INP, OUP, INO, OUO) \quad (3.3b)$$

$$INP = (inp_j^5)_{j5}^{\in J5} \quad (3.3c) \quad OUP = (oup_j^6)_{j6}^{\in J6} \quad (3.3d) \quad INO = (ino_j^7)_{j7}^{\in J7} \quad (3.3e) \quad OUO = (ouo_j^8)_{j8}^{\in J8} \quad (3.3f)$$

Описание на съставен елемент "Сегменти и Групи" (SG). Това е обединение на множества от различни типове сегменти и групи, като в него се съдържат разложени на съответни съставящи планираните за реализиране процеси и операции под формата на структурирани информационни последователности. В съответствие с горното можем да запишем:

$$SG = [INS_i^9, OUS_i^{10}, ING_i^{11}, OUG_i^{12}]_{i(9\div12)}^{\in I(9\div12)} \quad (3.4a) \qquad SG = f(INS, OUS, ING, OUG) \quad (3.4b)$$

$$INS = (ins_j^9)_{j9}^{\in J9} \quad (3.4c) \quad OUS = (ous_j^{10})_{j10}^{\in J10} \quad (3.4d) \quad ING = (ing_j^{11})_{j11}^{\in J11} \quad (3.4e) \quad OUG = (oug_j^{12})_{j12}^{\in J12} \quad (3.4f)$$

Описание на съставен елемент "Мрежови Управления" (NC). Това е обединението на множества от различни типове мрежови операционни системи (NOS), мрежови приложни програми (NAP), мрежови библиотечни програми (NLP) и мрежови драйверни програми (NDP). Този съставен елемент определя позицията на CPS в определени мрежови структури, зависимостта й от управляващите server/host/mainframe/domain системи и съответната й степен на "интелигентност". В съответствие с горното можем да запишем:

$$NC = [NOS_i^{13}, NAP_i^{14}, NLP_i^{15}, NDP_i^{16}]_{i(13\div16)}^{\in I(13\div16)} \quad (3.5a) \qquad NC = f(NOS, NAP, NLP, NDP) \quad (3.5b)$$

$$NOS = (nos_j^{13})_{j13}^{\in J13} \quad (3.5c) \quad NAP = (nap_j^{14})_{j14}^{\in J14} \quad (3.5d) \quad NLP = (nlp_j^{15})_{j15}^{\in J15} \quad (3.5e) \quad NDP = (ndp_j^{16})_{j16}^{\in J16} \quad (3.5f)$$

Описание на съставен елемент "Протоколни Управления" (PC). Това е обединението на множества от различни типове протоколи, включващи в конкретния случай трансмисионен контролен протокол (TCP), интернет предаващ протокол (IP), потребителски информационен (дейтаграмен) протокол (UDP) и контролиращ съобщенията протокол (ICMP). Този съставен елемент определя позицията на CPS в определени протоколни взаимодействия, свързани с предаване и преобразуване на информационни последователности, имащи решаващо отношение към информационната сигурност на мрежови структури и зависимостта на защитната система от управляващите server/host/mainframe/domain системи. В съответствие с горното можем да запишем:

$$PC = [TCP_i^{17}, IP_i^{18}, UDP_i^{19}, ICMP_i^{20}]_{i(17\div20)}^{\in I(17\div20)} \quad (3.6a) \qquad PC = f(TCP, IP, UDP, ICMP) \quad (3.6b)$$

$$TCP = (tcp_j^{17})_{j17}^{\in J17} \quad (3.6c) \quad IP = (ip_j^{18})_{j18}^{\in J18} \quad (3.6d) \quad UDP = (udp_j^{19})_{j19}^{\in J19} \quad (3.6e) \quad ICMP = (icmp_j^{20})_{j20}^{\in J20} \quad (3.6f)$$

Описание на съставен елемент "Хардуерни Средства" (HT). Това е обединението на множества от различни типове интегрирани процесорни системи (IPS), групирани по степен на интеграция и по функции, дискретни активни системи (DAS), дискретни пасивни системи (DPS), електромеханични системи (EMS). Елементът 'хардуерни средства' трябва да се разглежда в неразривно единство с елемента 'софтуерни средства'. Записът е следният:

$$HT = [IPS_i^{21}, DAS_i^{22}, DPS_i^{23}, EMS_i^{24}]_{i(21\div24)}^{\in I(21\div24)} \quad (3.7a) \qquad HT = f(IPS, DAS, DPS, EMS) \quad (3.7b)$$

$$IPS = (ips_j^{21})_{j21}^{\in J21} \quad (3.7c) \quad DAS = (das_j^{22})_{j22}^{\in J22} \quad (3.7d) \quad DPS = (dps_j^{23})_{j23}^{\in J23} \quad (3.7e) \quad EMS = (ems_j^{24})_{j24}^{\in J24} \quad (3.7f)$$

Описание на съставен елемент "Програмни Средства" (ST). Това е обединението на множества от различни типове локални операционни системи (LOS), локални приложни програми (LAP), локални библиотечни програми (LLP), локални драйверни програми (LDP). Това е най-сложният съставен елемент. Той отразява изключително сложните взаимоотношения между параметрите и характеристиките на програмното осигуряване на една CPS. Записът е:

$$ST = [LOS_i^{25}, LAP_i^{26}, LLP_i^{27}, LDP_i^{28}]_{i(25\div28)}^{\in I(25\div28)} \quad (3.8a) \qquad ST = f(LOS, LAP, LLP, LDP) \quad (3.8b)$$

$$LOS = (los_j^{25})_{j25}^{\in J25} \quad (3.8c) \quad LAP = (lap_j^{26})_{j26}^{\in J26} \quad (3.8d) \quad LLP = (llp_j^{27})_{j27}^{\in J27} \quad (3.8e) \quad LDP = (ldp_j^{28})_{j28}^{\in J28} \quad (3.8f)$$

Описание на съставен елемент "Диалогови Средства" (DT). Това е обединението на множества от различни типове екранни диалогови системи (SDS), гласови диалогови системи (VDS), (smart) картови диалогови системи (CDS) и биометрични диалогови системи (BDS). Инженерно-психологическите и ергономическите аспекти на диалоговия процес в разумен компромис с техническите възможности са основата, на която се създава този съставен елемент. Той е най-трудно представимият, тъй като е свързан с явления и процеси на висшата нервна дейност на човека-оператор, на строежа и

функционирането на човешкото око, на фотометричните, нефотометричните и електрофизиологичните параметри и характеристики на процеса на създаване, възпроизвеждане и възприемане на информацията. Елементът има следния запис:

$DT = [SDS_i^{29}, VDS_i^{30}, CDS_i^{31}, BDS_i^{32}]_{i(29\div32)}^{\in I(29\div32)}$ (3.9a) $DT = f(DSD, VDS, CDS, BDS)$ (3.9b)

$SDS = (sds_j^{29})_{j29}^{\in J29}$ (3.9c) $VDS = (vds_j^{30})_{j30}^{\in J30}$ (3.9d) $CDS = (cds_j^{31})_{j31}^{\in J31}$ (3.9e) $BDS = (bds_j^{32})_{j32}^{\in J32}$ (3.9f)

Описание на съставен елемент "Периферен Хардуер" (PH). Това е обединението на множества от различни типове входни устройства (InD), изходни устройства (OuD), входно-изходни устройства (IoD), трансформиращо - преобразуващи устройства (TCD). Чрез този съставен елемент се отразяват логическите и функционални отношения на CPS с нейните периферни (подчинени) устройства и системи. От своя страна, съставящите типове могат да се разглеждат като съставени от подмножества, групирани в зависимост от средата, в която се осъществява съответната операция. Така имаме съвкупности от оптически, магнитни, хартиени и други устройства, чиито математически запис е:

$PH = [InD_i^{33}, OuD_i^{34}, IOD_i^{35}, TCD_i^{36}]_{i(33\div36)}^{\in I(33\div36)}$ (3.10a) $PH = f(Ind, OuD, IOD, TCD)$ (3.10b)

$InD = (ind_j^{33})_{j33}^{\in J33}$ (3.10c) $OuD = (oud_j^{34})_{j34}^{\in J34}$ (3.10d) $IOD = (iod_j^{35})_{j35}^{\in J35}$ (3.10e) $TCD = (tcd_j^{36})_{j36}^{\in J36}$ (3.10f)

Описание на съставен елемент "Периферен Софтуер" (PS). Това е обединението на множества от различни типове периферни операционни системи (POS), периферни приложни програми (PAP), периферни библиотечни програми (PLP), периферни драйверни програми (PDP). Те съществуват в много голямо разнообразие и разновидности, но общото, което ги свързва е, че 'периферният софтуер' трябва да се разглежда в неразривно единство с 'периферния хардуер'. При тези предположения формалният запис е:

$PS = [POS_i^{37}, PAP_i^{38}, PLP_i^{39}, PDP_i^{40}]_{i(37\div40)}^{\in I(37\div40)}$ (3.11a) $PS = f(POS, PAP, PLP, PDP)$ (3.11b)

$POS = (pos_j^{37})_{j37}^{\in J37}$ (3.11c) $PAP = (pap_j^{38})_{j39}^{\in J39}$ (3.11d) $PLP = (plp_j^{39})_{j39}^{\in J39}$ (3.11e) $PDP = (pdp_j^{40})_{j40}^{\in J40}$ (3.11f)

*Основни елементи на оптимизационния процес.* На този етап от разглежданията е необходимо да определим основните елементи на оптимизационния процес, които служат за оценка и сравнение на различните варианти, съответно на различния състав, структура и конфигурация на моделите, съставящи общия модел на CPS, като заедно с това, тези основни елементи ще изпълняват задачи по формализираното описание на процесите в системата:

Параметри на оптимизационния процес (POP). Съвкупността от различните свойства на CPS като реален обект на оптимизация могат да бъдат представени чрез съответни параметри на оптимизационен процес. По-горе беше показано от колко много съставни части е изградена съвременната защитна система. Логичен е изводът, че в този случай параметрите наброяват стотици и е практически невъзможно да се извърши оптимизационен процес по всички тях, затова е нужно параметрите да бъдат групирани, да бъдат обобщени и в крайна сметка редуцирани до едно разумно количество, което да включва най-важните от тях.

Критерии на оптимизационния процес (COP). При наличието на една група от параметри на оптимизационен процес, които представляват най-важните свойства на CPS, може да се предложи те да бъдат обединени от една-единствена оценка, която има някаква числена стойност. Това по същество представлява единен критерий за оптималност (UCO). Но тъй като е невъзможно да бъде определен един-единствен критерий, то вместо него ще имаме поредица от критерии на оптимизационен процес, всеки от които ще характеризира отделно свойство на съставния елемент. По такъв начин в съответствие с направената формализация можем да запишем: $UCO = (COP_i^{(1\div40)})_{i(1\div40)}^{\in I(1\div40)}$ (3.12a) $COP_i^{(1\div40)} = (POP_j^{(1\div40)})_{j(1\div40)}^{\in J(1\div40)}$ (3.12b). Ако предположим, че са установени някакви други условия за функциониране ($CNF$) на десетелементната система, то единният критерий за оптималност ще бъде функция на десетте елемента на CPS, т.е. $UCO = f(FT, PO, SG, NC, PC, HT, ST, DT, PH, PS) \rightarrow CNF = const$ (3.13).

Целева функция на оптимизационния процес (TFOP). Тя е предназначена за глобална оценка на изследвания обект (в случая CPS) по отношение на предназначение и поставени задачи. Целевата функция има за аргумент единния критерий за оптималност, като съгласно основната концепция за модела на защитната система в единния критерий са отразени всички активни (управляващи) входни данни. Наличието на пасивни (неуправляващи) входни данни, участвуващи като аргумент в целевата функция, може да се отрази като множество от ограничаващи фактори (RSF), обединени в понятието

единно ограничение за оптималността ( URO ). Следователно можем да запишем: $URO = (RSF_i^{(1\div 40)})_{i(1\div 40)}^{\in I\,(1\div 40)}$ (3.14). В крайна сметка символичният запис на глобалната оценка на изследвания обект, изразена чрез целевата функция на оптимизационен процес е: $TFOP = f(UCO, URO)$ (3.15a) $TFOP = f(COP_i^{(1\div 40)}, RSF_i^{(1\div 40)})_{i(1\div 40)}^{\in I\,(1\div 40)}$ (3.15b) при което (3.15a) е съкратен запис, а (3.15b) е пълен запис.

*Основни етапи на оптимизационния процес.* Имайки предвид описаното по-горе и определени постановки в [2, 7, 10, 11], оптимизационният процес може да се раздели на пет етапа:

Задаване на параметри на оптимизационния процес за съставните елементи. Определят се критериите на оптимизационния процес и ограничителните условия на функциониране, след което се определя целевата функция на оптимизационния процес за всеки от съставните елементи. Всичко това може да се изрази като: $POP_i^{(1\div 40)}, COP_i^{(1\div 40)}, RSF_i^{(1\div 40)}, TFOP_i^{(1\div 40)}$ (3.16).

Опростяване на оптимизационния процес за съставните елементи. В условията на горните допускания се получава ситуация, при която всеки един съставен елемент от CPS се характеризира с много голям брой параметри на оптимизационен процес, а това означава и много голям брой варианти, които трябва да изследва оптимизационния процес. В общия случай оптимизационната задача е практически неразрешима [22, 90, 240, 241, 242, 244, 263, 270, 276, 280, 281, 287, 290] и поради това е необходимо да се въведат опростявания, като всеки един компромис с реалния обект е внимателно преценен. Опростяването не трябва да става и за сметка на параметрите, които ще придобият решаващо значение при свързването на съставните елементи в един общ обект на оптимизация;

Задаване на параметрите на оптимизационен процес за CPS. Основната идея на този етап е, че показаните на (3.16) съвкупности, в съответствие с описаното по-горе, се конкретизират и опростяват по такъв начин, че те да бъдат валидни за общия обект на оптимизация – CPS, при това изборът на параметри на оптимизационен процес за CPS трябва да се определя от тяхната функционална съпричастност с поставената оптимизационна задача;

Опростяване на оптимизационния процес за CPS. Мерките, които трябва да се вземат, за да се опрости общия оптимизационен процес, отчитайки направените досега разглеждания, могат да се групират по следния начин: 1) Ограничаване на параметрите на оптимизационния процес за съставните елементи на CPS. Това означава: 1. Множеството от различни типове функции и задачи да се стесни до конкретен тип; 2. Множеството различни типове съставни елементи да се стесни до конкретен тип; 3. Множеството от различни типове характерни компоненти да се стесни до конкретен тип; 4. Множеството от различни типове параметри, характерни за конкретен тип характерен компонент, да се стесни до конкретен тип параметри. 2) Опростяване на оптимизационната задача за CPS. Това означава: 1. Оптимизацията на CPS се разделя на отделни подоптимизации [18, 19, 33, 37], провеждани в даден съставен елемент, като се предполага, че в останалите съставни елементи няма изменения; 2. Подоптимизациите се извършват по отношение на избрани от цялото множество критерии на оптимизационен процес, които ще означаваме като $COP_{sel}^{xxx}$; 3. Избраният критерий на оптимизационен процес, който е по същество многомерен вектор, се опростява, като се заменя с едномерна величина.

Реализиране на оптимизационен процес за CPS. При този етап се извършва оптимизационен процес за CPS като процес на търсене на екстремум на една целева функция по отношение на избрани критерии: 1) Обща формула на оптимизационен процес за CPS. Ако приемем, че типът на екстремума на целевата функция на оптимизационен процес е максимум и отчитайки (3.13), (3.14) и (3.15), можем да запишем:

$$TFOP^{max} = TFOP_i^{max}(UCO_i^{max}, URO_i^{min}) =$$

$$= UCO^{max} \begin{Bmatrix} FT\left[INF_i^{ct1.1}\left(COP_i^{inf}\right), OUF_i^{ct1.2}\left(COP_i^{ouf}\right), INT_i^{ct1.3}\left(COP_i^{int}\right), OUT_i^{ct1.4}\left(COP_i^{out}\right)\right], \\ PO\left[INP_i^{ct2.1}\left(COP_i^{inp}\right), OUP_i^{ct2.2}\left(COP_i^{oup}\right), INO_i^{ct2.3}\left(COP_i^{ino}\right), OUO_i^{ct2.4}\left(COP_i^{uo}\right)\right], \\ SG\left[INS_i^{ct3.1}\left(COP_i^{ins}\right), OUS_i^{ct3.2}\left(COP_i^{ous}\right), ING_i^{ct3.3}\left(COP_i^{ing}\right), OUG_i^{ct3.4}\left(COP_i^{oug}\right)\right], \\ NC\left[NOS_i^{ct4.1}\left(COP_i^{nos}\right), NAP_i^{ct4.2}\left(COP_i^{nap}\right), NLP_i^{ct4.3}\left(COP_i^{nlp}\right), NDP_i^{ct4.4}\left(COP_i^{ndp}\right)\right], \\ PC\left[TCP_i^{ct5.1}\left(COP_i^{tcp}\right), IP_i^{ct5.2}\left(COP_i^{ip}\right), UDP_i^{ct5.3}\left(COP_i^{udp}\right), ICMP_i^{ct5.4}\left(COP_i^{icmp}\right)\right], \\ HT\left[IPS_i^{ct6.1}\left(COP_i^{ips}\right), DAS_i^{ct6.2}\left(COP_i^{das}\right), DPS_i^{ct6.3}\left(COP_i^{dps}\right), EMS_i^{ct6.4}\left(COP_i^{ems}\right)\right], \\ ST\left[LOS_i^{ct7.1}\left(COP_i^{los}\right), LAP_i^{ct7.2}\left(COP_i^{lap}\right), LLP_i^{ct7.3}\left(COP_i^{llp}\right), LDP_i^{ct7.4}\left(COP_i^{ldp}\right)\right], \\ DT\left[SDS_i^{ct8.1}\left(COP_i^{sds}\right), VDS_i^{ct8.2}\left(COP_i^{vds}\right), CDS_i^{ct8.3}\left(COP_i^{cds}\right), BDS_i^{ct8.4}\left(COP_i^{bds}\right)\right], \\ PH\left[IND_i^{ct9.1}\left(COP_i^{ind}\right), OUD_i^{ct9.2}\left(COP_i^{oud}\right), IOD_i^{ct9.3}\left(COP_i^{iod}\right), TCD_i^{ct9.4}\left(COP_i^{tcd}\right)\right], \\ PS\left[POS_i^{ct10.1}\left(COP_i^{pos}\right), PAP_i^{ct10.2}\left(COP_i^{pap}\right), PLP_i^{ct10.3}\left(COP_i^{plp}\right), PDP_i^{ct10.4}\left(COP_i^{pdp}\right)\right], \end{Bmatrix},$$

$$URO^{min} \begin{Bmatrix} FT\left[INF_i^{ct1.1}\left(RSF_i^{inf}\right), OUF_i^{ct1.2}\left(RSF_i^{ouf}\right), INT_i^{ct1.3}\left(RSF_i^{int}\right), OUT_i^{ct1.4}\left(RSF_i^{out}\right)\right], \\ PO\left[INP_i^{ct2.1}\left(RSF_i^{inp}\right), OUP_i^{ct2.2}\left(RSF_i^{oup}\right), INO_i^{ct2.3}\left(RSF_i^{ino}\right), OUO_i^{ct2.4}\left(RSF_i^{uo}\right)\right], \\ SG\left[INS_i^{ct3.1}\left(RSF_i^{ins}\right), OUS_i^{ct3.2}\left(RSF_i^{ous}\right), ING_i^{ct3.3}\left(RSF_i^{ing}\right), OUG_i^{ct3.4}\left(RSF_i^{oug}\right)\right], \\ NC\left[NOS_i^{ct4.1}\left(RSF_i^{nos}\right), NAP_i^{ct4.2}\left(RSF_i^{nap}\right), NLP_i^{ct4.3}\left(RSF_i^{nlp}\right), NDP_i^{ct4.4}\left(RSF_i^{ndp}\right)\right], \\ PC\left[TCP_i^{ct5.1}\left(RSF_i^{tcp}\right), IP_i^{ct5.2}\left(RSF_i^{ip}\right), UDP_i^{ct5.3}\left(RSF_i^{udp}\right), ICMP_i^{ct5.4}\left(RSF_i^{icmp}\right)\right], \\ HT\left[IPS_i^{ct6.1}\left(RSF_i^{ips}\right), DAS_i^{ct6.2}\left(RSF_i^{das}\right), DPS_i^{ct6.3}\left(RSF_i^{dps}\right), EMS_i^{ct6.4}\left(RSF_i^{ems}\right)\right], \\ ST\left[LOS_i^{ct7.1}\left(RSF_i^{los}\right), LAP_i^{ct7.2}\left(RSF_i^{lap}\right), LLP_i^{ct7.3}\left(RSF_i^{llp}\right), LDP_i^{ct7.4}\left(RSF_i^{ldp}\right)\right], \\ DT\left[SDS_i^{ct8.1}\left(RSF_i^{sds}\right), VDS_i^{ct8.2}\left(RSF_i^{vds}\right), CDS_i^{ct8.3}\left(RSF_i^{cds}\right), BDS_i^{ct8.4}\left(RSF_i^{bds}\right)\right], \\ PH\left[IND_i^{ct9.1}\left(RSF_i^{ind}\right), OUD_i^{ct9.2}\left(RSF_i^{oud}\right), IOD_i^{ct9.3}\left(RSF_i^{iod}\right), TCD_i^{ct9.4}\left(RSF_i^{tcd}\right)\right], \\ PS\left[POS_i^{ct10.1}\left(RSF_i^{pos}\right), PAP_i^{ct10.2}\left(RSF_i^{pap}\right), PLP_i^{ct10.3}\left(RSF_i^{plp}\right), PDP_i^{ct10.4}\left(RSF_i^{pdp}\right)\right], \end{Bmatrix} \quad (3.17)$$

При условие, че $RSF_i$ са постоянни и $COP_i$ са представени обобщено като функции на отделните типове елементи на CPS, то (3.17) има вида:

$$TFOP^{max} = TFOP_i^{max} COP_i^{max}(ft_i, po_i, sg_i, nc_i, pc_i, ht_i, st_i, dt_i, ph_i, ps_i) \quad (3.18)$$

Тогава екстремумът (в случая максимум) на $COP_i$ може да се изрази така:

=

$$FT\left[ ft_i \left(\inf_{j1} \in INF \ , ouf_{j2} \in OUF \ , \mathrm{int}_{j3} \in INT \ , out_{j4} \in OUT \ \right)\right],$$

$$PO\left[ po_i \left(inp_{j1} \in INP \ , oup_{j2} \in OUP \ , ino_{j3} \in INO \ , ouo_{j4} \in OUO \ \right)\right],$$

$$SG\left[ sg_i \left(ins_{j1} \in INS \ , ous_{j2} \in OUS \ , ing_{j3} \in ING \ , oug_{j4} \in OUG \ \right)\right],$$

$$NC\left[ nc_i \left(nos_{j1} \in NOS \ , nap_{j2} \in NAP \ , nlp_{j3} \in NLP \ , ndp_{j4} \in NDP \ \right)\right],$$

$$PC\left[ pc_i \left(tcp_{j1} \in TCP \ , ip_{j2} \in IP \ , udp_{j3} \in UDP \ , icmp_{j4} \in ICMP \ \right)\right], \qquad (3.19)$$

$$HT\left[ ht_i \left(ips_{j1} \in IPS \ , das_{j2} \in DAS \ , dps_{j3} \in DPS \ , ems_{j4} \in EMS \ \right)\right],$$

$$ST\left[ st_i \left(los_{j1} \in LOS \ , lap_{j2} \in LAP \ , llp_{j3} \in LLP \ , ldp_{j4} \in LDP \ \right)\right],$$

$$DT\left[ dt_i \left(sds_{j1} \in SDS \ , vds_{j2} \in VDS \ , cds_{j3} \in CDS \ , bds_{j4} \in BDS \ \right)\right],$$

$$PH\left[ ph_i \left(ind_{j1} \in IND \ , oud_{j2} \in OUD \ , iod_{j3} \in IOD \ , tcd_{j4} \in TCD \ \right)\right],$$

$$PS\left[ ps_i \left(pos_{j1} \in POS \ , pap_{j2} \in PAP \ , plp_{j3} \in PLP \ , pdp_{j4} \in PDP \ \right)\right]$$

Конкретните стъпки на оптимизационния процес за CPS се получават от прилагането на (3.17) и (3.19) към оптимизационната задача; 2) Методи за осъществяване на оптимизационния процес за CPS. Търсенето на екстремум на целевата функция може да стане по много начини, всеки от които има своите недостатъци. Позовавайки се на [22, 24, 27, 33, 38, 44, 46] ще разгледаме само два от тях, с най-голямо приложение: 1. Метод на Монте Карло. Той се използва тогава, когато е налице достатъчна информация за решенията на оптимизационната задача, свързани с избран критерий на оптимизационен процес, при което е възможно да се изчисли вероятността за включване на оптималното решение в кръга от случайно избрани решения [9, 42, 162, 170, 208]; 2. Метод на градиента. Той е подходящ за използване в случаите, когато имаме минимална информация за възможните стойности на решенията на оптимизационната задача. При него търсенето на екстремума (минимизиране или максимизиране), става в посоката на най-голямо изменение (в посока на градиента) [5, 7, 19, 33, 45].

*Алгоритъм за оптимизационен процес.* В резултат на всичко казано до тук, може да се предложи едно обобщение, което има формата на блокова схема на алгоритъм.

МЕТОДИКА ЗА СИМУЛАЦИОННО МОДЕЛИРАНЕ С ОПТИМИЗАЦИОНЕН ПРОЦЕС. Настоящата методика трябва да се разглежда като резултат от направените по-горе разглеждания, отнасящи се до методиката за симулационно моделиране и до оптимизационен процес за CPS по определени критерии. Тя може да се представи под формата на резултантен граф, като с помощта на отделни елементи, свързани помежду си, се показва последователността от стъпки, които образуват резултантната методика за симулационно моделиране с оптимизационен процес. Означенията, които могат да се използват, имат следния смисъл: SM – симулационно моделиране; OP – оптимизационен процес; 1, 2, 3, 4, и 5, поставени до горните две съкращения, означават номера на етапа, към който принадлежи стъпката; след знака "-" се записват цифри, които представляват поредния номер на всяка стъпка. Методиката за симулационно моделиране и оптимизационен процес е съставена от пет етапа:

*Подготовка за симулационно моделиране.* Този етап съдържа 5 последователни и 40 паралелни стъпки. Последователните стъпки включват дефинирането на проблема (SM1-1), избор на метода за изследване (SM1-2), прогноза за възможните решения (SM1-3), избор на база за изследване (SM1-4) и определяне на необходимите ресурси (SM1-5), а паралелните – задаване на параметрите на оптимизационния процес от (OP1-1.1) до (OP1-10.1), на критериите на оптимизационния процес от (OP1-1.2) до (OP1-10.2), на ограничаващите фактори от (OP1-1.3) до (OP1-10.3) и на целевата функция на оптимизационния процес от (OP1-1.4) до (OP1-10.4) за съставните елементи на CPS. Смисълът на паралелните стъпки, е че става възможно да се провеждат оптимизационни процеси за отделните съставни елементи, като се запазва алгоритъма на резултантната методика в неговата степен на детайлност и обобщение.

*Формиране на съставни модели.* В етапа са включени две групи от паралелни стъпки. Първата група съдържа 60 стъпки, които включват създаване на концептуален модел от (SM2-6.1) до (SM2-15.1), определяне на критериите за оценка от (SM2-6.2) до (SM2-15.2), определяне на съотношенията между критериите за оценка от (SM2-6.3) до (SM2-15.3), съставяне на моделиращ алгоритъм от (SM2-6.4) до

(SM2-15.4), реализиране на работен модел от (SM2-6.5) до (SM2-15.5) и проверка за достоверността на модела от (SM2-6.6) до (SM2-15.6), за съответните съставни елементи. Втората група съдържа 40 стъпки, които включват конкретизиране на параметрите на оптимизационния процес от (OP2-11.1) до (OP2-20.1), на критериите на оптимизационния процес от (OP2-11.2) до (OP2-20.2), на ограничаващите фактори от (OP2-11.3) до (OP2-20.3) и на целевата функция на оптимизационния процес от (OP2-11.4) до (OP2-20.4) за съставните елементи.

*Експерименти със съставни модели.* Етапът съдържа 50 паралелни и 4 последователни стъпки. Паралелните стъпки включват планиране на експериментите от (SM3-16.1) до (SM3-25.1), задаване на входните данни за параметрите на оптимизационния процес от (SM3-16.2) до (SM3-25.2), проиграване на моделите и събиране на резултатите от (SM3-16.3) до (SM3-25.3), анализ на резултатите от (SM3-16.4) до (SM3-25.4) и оценки и изводи от (SM3-16.5) до (SM3-25.5) за проведените експерименти със съставните модели. Ако е нужно, следвайки направените изводи и оценки, се правят необходимите корекции в моделите. Последователните стъпки включват задаване на параметрите на оптимизационния процес (OP3-21), на критериите на оптимизационния процес (OP3-22), на ограничаващите фактори (OP3-23) и на целевата функция (OP3-24) за оптимизационния процес на CPS.

*Формиране на общ модел.* Съдържа 6 последователни и 90 паралелни стъпки. Последователните стъпки съдържат създаване на концептуален модел (SM4-26), определяне на критериите за оценка (SM4-27), определяне на съотношенията между критериите за оценка (SM4-28), съставяне на моделиращ алгоритъм (SM4-29), реализиране на работен модел (SM4-30) и проверка за достоверност (SM4-31) за общия модел на изследвания обект – CPS. Паралелните стъпки съдържат избор на тип съставен елемент от (OP4-25.1) до (OP4-34.1), на тип характерен компонент от (OP4-25.2) до (OP4-34.2), на тип СС от (OP4-25.3) до (OP4-34.3), на тип параметър на оптимизационен процес от (OP4-25.4) до (OP4-34.4), на тип критерий на оптимизационния процес от (OP4-25.5) до (OP4-34.5) и на тип целева функция на оптимизационния процес от (OP4-25.6) до (OP4-34.6), след което се прави избор на подоптимизация за определен характерен компонент по определена целева функция на оптимизационен процес от (OP4-25.7) до (OP4-34.7) и избор на едномерна стойност на критериите на оптимизационен процес от (OP4-25.8) до (OP4-34.8), като по такъв начин се подготвя последната поредица от стъпки – търсене на екстремум за избраната целева функция на оптимизационен процес при фиксирани условия на функциониране, така че ограничаващите фактори да са константа - от (OP4-25.9) до (OP4-34.9). При този етап, като се използва принципа на подоптимизацията и подхода на множествената декомпозиция, става възможно оптимизационният процес за CPS да бъде конкретизиран и опростен до необходимото ниво в зависимост от поставените оптимизационни задачи.

*Експерименти с общия модел.* В този етап са включени общо 21 стъпки. Първите 4 са последователни и включват определяне на опростена целева функция на оптимизационен процес (OP5-35), явяваща се резултат от предходните стъпки, определяне на вида на екстремума на опростената целева функция на оптимизационен процес (OP5-36), определяне на вида на екстремума на опростения критерий на оптимизационен процес (OP5-37), определяне на условията на функциониране, така че ограничаващите фактори да са константа (OP5-38). Следващите 17 стъпки са организирани в два цикъла: външен и вътрешен. Външният цикъл включва избиране на група от възможни стойности за опростените параметри за оптимизационен процес (OP5-39), избиране на група от възможни стойности за опростените критерии за оптимизационен процес (OP5-40), планиране на експериментите с общия модел (SM5-32), задаване на входните данни за общия модел (SM5-33), избиране на една стойност за опростените критерии на оптимизационен процес (OP5-41), и за опростените параметри на оптимизационен процес (OP5-42), (от тук започва вътрешният цикъл), проиграване на модела и събиране на резултатите (SM5-34), анализ на резултатите (SM5-35), оценки и изводи (SM5-36), изчисляване на опростената целева функция на оптимизационен процес за CPS (OP5-43), изчисляване на разликата между стойностите на зададената в (OP3-24) и опростената целева функция (OP5-43) – това е (OP5-44), проверяване степента на точност на получения резултат за опростената целева функция, като се проверява MIN дали е по-малко от $EPSILON_{req}$ (което се задава в (OP5-45)); ако не е по-малко се избира нова стойност за опростения параметър на оптимизационен процес (OP5-46) и вътрешният цикъл се затваря; ако е по-малко, даденият параметър на оптимизационен процес се определя като максимален и се фиксира за текущата стойност на критерия (OP5-47), след което се проверява, дали е изчерпана групата от възможни стойности за опростените критерии (OP5-48), ако не е изчерпана, се избира нова стойност за опростения критерий (OP5-49) и външния цикъл се затваря; ако групата е изчерпана, се отива на последната стъпка – фиксиране на получените стойности на

параметрите на оптимизационен процес на CPS, за които целевата функция е максимална по отношение на определени критерии (OP5-50).

ПОДГОТВИТЕЛЕН ЕТАП ЗА СЪЗДАВАНЕ НА СИМУЛАЦИОННИ МОДЕЛИ.

*Дефиниране на проблема (SM1-1).* Изследването на CPS с цел оптимизация на параметрите по определени критерии ще се извърши по методиката, изложена по-горе. Голямото многообразие от варианти за изследване на защитната система е резултат и от големия брой типове съставни елементи, които могат да се намалят, като се групират по следния нов начин: 1) FT' – представляващи обединението на трите входни натоварвания на компютърната защитната система, т.е. FT'=FT+PO+SG ; 2) HT' – представляващи обединението на трите хардуерни съставни елементи на CPS, т.е. HT'=HT+DT+PH ; 3) ST' – представляващи обединението на четирите програмни съставни елемента на CPS, т.е. ST'=NC+PC+ST+PS. Следвайки предложената методика, в тази стъпка трябва да определим конкретните задачи за изследване на CPS: 1) За конкретен тип функции и задачи ($FT_i$ =const ) да се изследват характеристиките на CPS при изменение на състава и структурата за HT' и ST' с цел оптимизация на конкретни нейни параметри; 2) За конкретен тип управление ($MNG_i$ =const ) да се изследват характеристиките на CPS при изменение на типа на FT', с цел определяне на оптималния тип FT' за конкретна структура и състав; 3) За конкретна комбинация от конкретни типове състав и структура [CMP(HT'&ST'=const)&STR(HT'&ST'=const)] да се определи конкретен тип FT', които да са достатъчно универсални за изпълнение от CPS.

Определяне на функциите и задачите. Основното изискване към тях е да могат да се разделят на отделни части, позволяващи едновременно (паралелно) изпълнение, или отделните функции и задачи да принадлежат към съвкупност от взаимосвързани задачи, които се решават паралелно-последователно във времето. Естествено това изискване се отнася преди всичко за алгоритъма на изпълнение на такива функции и задачи. Най-често този алгоритъм се описва чрез граф, който представлява графичен запис на последователността на отделните стъпки от обработката на функцията и задачата във времето и пространството. Така че ограниченията към отделната функция и задача се правят най-лесно като ограничения към нейния граф. Известни са различни видове на графа за определени задачи [41, 44, 51]. В нашия случай, изхождайки от целите на настоящата дисертация и позовавайки се на [44, 49], можем да определим, че необходимият ни граф е от вида дискретно-разпределен. Съществен момент е обстоятелството, че графът отразява не структурата на конкретната функция и задача, а структурата на избрания модел за решение, поради което графът предопределя всички възможни варианти на изпълнение на функцията и задачата. Един примерен дискретно-разпределен граф ( DDG ), при който изпълнението на програмата е разделено на дискрети от време, изменящи се по хоризонтала, като във всеки дискрет от време се осъществяват няколко едновременно изпълнявани операции, изменящи се по вертикала. Този граф се състои от възли и дъги, които са насочени. По такъв начин, всеки участък от програмата, изпълняваща функция и задача, който може да се представи чрез възел в дискретно разпределения граф, ще представлява един програмен елемент ( PE ). Дъгите, които са насочени към/от възела, могат да бъдат три вида: 1) $a_{i,m}$ – представляват входните данни за $PE_i$ (i – номер на програмния елемент, m – пореден номер на входните данни в програмата); 2) $b_{j,i}$ – представляват изходните резултати от работата на $PE_i$ , които са предназначени за $PE_j$ като негови входни данни; 3) $c_i$ – представляват резултати от работата на $PE_i$ , които са окончателни за дадена функция и задача. Освен това дъгите показват наличието на програмни обвързаности между два програмни елемента, които се наричат междинни програмни връзки ( IPB ). Един $PE_j$ ще бъде изпълнен само, ако всички $PE_i$ , от които той зависи, са вече изпълнени и може да се осъществи междинна програмна връзка, отразена чрез дъгата $b_{j,i}$. Естествен е въпросът, дали избраният тип FT' е достатъчно универсален. Отговорът може да се намери в обширните изследвания на алгоритми с дискретно разпределени графи в [6, 11, 23, 32, 63, 73, 82, 98, 103, 106], където са направени многобройни математически доказателства за използването му в задачи като: 1) Сума и произведение от числови редове; 2) Числени методи за решаване на системи от: 1. Обикновени диференциални уравнения; 2. Линейни алгебрични уравнения; 3. Уравнения с частни производни и други; 3) Коефициенти на суми и произведения на степенни редове; 4) Приближено решаване на интеграли чрез преобразуванията на Фурие; 5) Рекурентни зависимости за деление; 6) Матрични, векторни анализи и други. При тези предпоставки трябва да

направим извода, че избраният метод на представяне на паралелно-последователни функции и задачи чрез дискретно разпределен граф е достатъчно универсален и може да бъде добра основа при определяне на подходящите състав и структура на конкретна CPS.

Определяне на ресурсите и взаимодействието им. Имайки предвид избрания тип FT', можем да направим предположения за ресурсите на CPS, които трябва да бъдат включени в структурата: (1) Процесор ( PRC ) – извършва обработката и изпълнението на програмния елемент. Обикновено е реализиран чрез микропроцесор, който може да бъде интегрален или съставен от няколко микропроцесорни секции. В зависимост от изчислителното натоварване на CPS броят на процесорите е различен. Режимите на работа на процесорите, като се има предвид определения тип функции и задачи, се определят преди всичко от отделните етапи на обработка, през които трябва да премине един програмен елемент: 1) Начална четяща операция (SRO) – четене на входните данни $a_{i,m}$; 2) Междинна четяща операция (IRO) – четене на междинните резултати $b_{j,l}$; 3) Обработваща операция (PRO) – обработка на програмния елемент; 4) Междинна записваща операция (IWO) – запис на крайните резултати $c_i$ от изпълнението на програмния елемент, които могат да се използват като междинна програмна връзка към други програмни елементи; 5) Пълна обработваща операция (CPO) – тя е обобщение на *PRO* и *IWO*; 6) Крайна записваща операция (FWO) – запис на крайните резултати от изпълнението на програмния елемент. (2) Памет ( MEM ) – която може да бъде: 1) Програмна памет ( $MEM^p$ ) – тя служи за съхраняване на управляващите програми на CPS. Тя може да бъде реализирана на постоянно запомнящо устройство или на оперативно запомнящо устройство. Във втория случай управляващите програми се зареждат чрез специална процедура по линията от управляващата система server/host/mainframe/domain или от локално разположено външно съхраняващо устройство. Освен това програмната памет може да съдържа програмна реализация на широк кръг (или всички) функции и задачи заедно с дискретно разпределения граф за тяхното изпълнение; 2) Работна памет ( $MEM^w$ ) – тя служи за съхраняване на всички междинни резултати, които могат да се използват като междинни програмни връзки, като може да бъде реализирана на полупроводниково оперативно запомнящо устройство или на бързо дисково запомнящо устройство. Ако процесорът е повече от един, тя може да бъде разделена на отделни части, които са пряко достъпни за всеки отделен процесор, при това броят на частите е равен на броя на процесорите; 3) Даннова памет ( $MEM^d$ ) – тя служи за съхраняване на входни данни и на крайни резултати от изпълнението на дадени програмни елементи. Тази памет може да бъде разделяна на части, които са пряко достъпни за отделен процесор, като броят на частите е равен на броя на процесорите. В зависимост от сложността на избраната структура за CPS, паметите в нея могат да бъдат управлявани чрез специално управляващо устройство памет (CDMEM), което в зависимост от натоварването да прерасне в специален процесор за управление на всички видове памети, както и за управление на съответните носители. (3) Периферия ( PRF ) – в нея се включват: 1) Входна периферия ( $PRF^{in}$ ) – в нея влизат всички устройства, осигуряващи входните данни за останалите ресурси на CPS; 2) Изходна периферия ( $PRF^{out}$ ) – в нея влизат всички устройства, осигуряващи извеждането на крайни резултати от изпълнението на отделен програмен елемент или на програмата, която решава отделна функция и задача; 3) Шинна периферия ( $PRF^{bus}$ ) – наименованието не е съвсем точно, но това е направено в интерес на класификацията. Има се предвид шинната система от връзки, която свързва всички отделни части на структурата на CPS. В нея са включени три функционални групи: адресни, даннови и управляващи връзки; и две разновидности: външна ( $PRF^{bus}_{external}$ ) и вътрешна ( $PRF^{bus}_{internal}$ ). 4) Управление ( MNG ) – в този ресурс се включват по същество ST'. Неговото предназначение е да осигури цялостно управление на отделните изчислителни процеси, както и на общото функциониране на CPS. Отличителна черта на този ресурс е неговото нефиксирано физическо местоположение, защото работата на CPS може да се координира от управляващи системи server/host/mainframe/domain или от управляващи програми, разположени в програмната памет. В първия случай имаме сложни правила за подчиненост на CPS по отношение на управляващите системи и затова на този етап ще считаме, че този ресурс е разположен в програмната памет. Освен това на този етап няма да се спираме на вътрешната структура на ST' с цел да не се получи прекалена детайлност, което ще затрудни създаването на симулационните модели. Сега можем да направим някои предположения за взаимодействието между ресурсите при функциониране на CPS в процеса на изпълнение на определения тип функции и задачи:

(1) Процесорът на CPS в зависимост от изчислителното натоварване може да бъде повече от един, при това, отчитайки характерните особености на изчислителния процес, множеството на процесорите може да бъде разделено на функционално специализирани групи, които съгласно [1, 8, 17, 20, 32, 37, 43, 47, 48, 53, 54, 61, 74, 96, 99, 133] могат да бъдат означени, например, по следния начин: 1) Ниво 7 от модела ISO/OSI (приложение – специализирани мрежови функции като файлов трансфер, виртуален терминал, електронна поща, файлови сървъри и други): $PRC^{TFTP}$ – TFTP (Trivial File Transfer Protocol) процесор; $PRC^{BOOTP}$ – BOOTP (Bootstrap Protocol) процесор; $PRC^{SNMP}$ – SNMP (Simple Network Management Protocol) процесор; $PRC^{FTP}$ – FTP (File Transfer Protocol) процесор; $PRC^{SMTP}$ – SMTP (Simple Mail Transfer Protocol) процесор; $PRC^{MIME}$ – MIME (Multipurpose Internet Mail Extensions) процесор; 2) Ниво 6 от модела ISO/OSI (представяне – форматиране на данните, конвертиране на буквените кодове и кодиране на данни): това ниво не се използва от пакета TCP/IP тъй като няма официално дефинирани засега протоколи и съответно няма специализирани процесори; 3) Ниво 5 от модела ISO/OSI (сесия – свързване и установяване на връзка с друг възел): това ниво не се използва от пакета TCP/IP тъй като няма официално дефинирани засега протоколи и съответно няма специализирани процесори; 4) Ниво 4 от модела ISO/OSI (транспортиране - осигурява предаването на данните): $PRC^{TCP}$ – TCP (Transmission Control Protocol) процесор; $PRC^{UDP}$ – UDP (User Datagram Protocol) процесор; 5) Ниво 3 от модела ISO/OSI (мрежа – насочване на пакетите с информация през различните мрежи): $PRC^{IP}$ – IP (Internet Protocol) процесор; $PRC^{ICMP}$ – ICMP (Internet Control Message Protocol) процесор; $PRC^{RIP}$ – RIP (Routing Information Protocol) процесор; $PRC^{OSPF}$ – OSPF (Open Shortest Path First) процесор; $BGP^{BGP}$ – BGP (Border Gateway Protocol) процесор; $PRC^{IGMP}$ – IGMP (Internet Group Management Protocol) процесор; 6) Ниво 2 от модела ISO/OSI (канал – трансфер на адресираните единици рамки и проверка за грешки): $PRC^{SLIP}$ – SLIP (Serial Line Internet Protocol) процесор; $PRC^{CSLIP}$ – CSLIP (Compressed Serial Line Internet Protocol) процесор; $PRC^{PPP}$ – PPP (Point-to-Point Protocol) процесор; $PRC^{ARP}$ – ARP (Address Resolution Protocol) процесор; $PRC^{RARP}$ – RARP (Reverse Address Resolution Protocol) процесор; $PRC^{MTU}$ – MTU (Maximum Transmission Unit) процесор; 7) Ниво 1 от модела ISO/OSI (физически – предаване на бинарните данни през комуникационната мрежа, като тук се включва подслоят на горното ниво Media Access Control (MAC)): $PRC^{802.3}$ – IEEE802.3 CSMA/CD процесор; $PRC^{802.4}$ – IEEE802.4 Token bus процесор; $PRC^{802.5}$ – IEEE802.5 Token Ring процесор; $PRC^{802.12}$ – IEEE802.12 Demand priority процесор. Изброените по-горе (на брой 24) основни типове процесори са почти максималното, което е необходимо за създаването на най-висококачествени и най-скъпи CPS; (2) Процесорите, когато са свободни, сигнализират за това на управлението, в резултат на което то им изпраща за изпълнение следващ програмен елемент. Възможно е да получат за обработка и междинен резултат от някой обработващ се вече програмен елемент, но това ще считаме за вътрешен процес. По същество, пред управлението се образува опашка от процесори, чакащи да получат програмен елемент за изпълнение. Дисциплината на обслужване на опашката може да се регламентира със система от приоритети; (3) Управлението осигурява обработката на функциите и задачите, имайки предвид съответния алгоритъм, който е описан във вид на дискретно разпределен граф, намиращ се в програмната памет. Предназначението на управлението може да се сведе до определяне на свободен (все още не взет за изпълнение) и изпълним (всички програмни елементи, от които той зависи чрез междинните програмни връзки са вече изпълнени) програмен елемент, който се изпраща за изпълнение към първия в опашката процесор; (4) Програмният елемент, който ще се обработва се намира в данновата памет, като той е пристигнал там или от програмната памет или от входната периферия. Освен това, броят на програмните елементи в даден момент от време е равен на броя на процесорите; (5) Последователността от действия, които извършва, например, k -тият процесор при изпълнение на i -тия програмен елемент са следните: 1) Разрешава SRO – ако програмният елемент $PE_i$ се намира извън данновата памет, той се прехвърля от програмната памет или входната периферия чрез шинната периферия; 2) Разрешава IRO – прочитат се междинните резултати, касаещи програмния елемент; 3) Разрешава PRO – изпълнява се самият програмен елемент $PE_i$ ; 4) Разрешава IWO – ако получените резултати имат отношение към други програмни елементи, те се записват в работната памет; 5) Разрешава FWO – ако получените резултати нямат отношение към други програмни елементи

чрез шинната периферия, те се изпращат в данновата памет, ако трябва да се групират, или се изпращат направо в изходната периферия; (6) Общото време за изпълнение на $PE_i$ от $PRC_k$ ще означим като $t_{i,k}^{pe}$ и то ще бъде равно на сумата от времената за изпълнение на отделните операции, към която се прибавя и времето $t_{i,k}^{wait}$ на чакане на $PRC_k$ на опашки за изпълнение на неговите заявки за начална четяща, междинна четяща, междинна записваща и крайна записваща операции. Формалният запис е следният: $t_{i,k}^{pe}=t_{i,k}^{sro}+t_{i,k}^{iro}+t_{i,k}^{pro}+t_{i,k}^{iwo}+t_{i,k}^{fwo}+t_{i,k}^{wait}$ (3.20); (7) Времето за изпълнение на IRO за $PE_i$ от $PRC_k$ е съставно и е равно на сумата от времената за изпълнение на отделните междинните програмни връзки, касаещи $PE_i$, т.е. $t_{i,k}^{iro}=\sum_{b=1}^{B} H_b t_{i,k,b}^{iro}$ (3.21) където: B - общ брой на междинните програмни връзки, касаещи $PE_i$; $H_b=1$, ако има междинни програмни връзки и $H_b=0$, ако няма междинни програмни връзки; (8) В по-нататъшното изложение индексите i и k, с цел по-голяма прегледност, ще бъдат изпускани на тези места, където няма да се допусне двусмислие; (9) Времената на чакане, които бяха споменати по-горе, се определят от два фактора: начинът на изпълнение на дискретно разпределения граф – търсене на свободен и изпълним програмен елемент; и начинът на съставяне на дискретно разпределения граф – той определя броя на междинните програмни връзки за всеки програмен елемент и за цялата функция и задача като цяло; (10) Изследването на алгоритмите за управление, се извършва в следната последователност: 1) Задава се множеството от програмни елементи, които ще представляват задания, възлагани на процесора за изпълнение; 2) Задаване на множеството от свободни и изпълними програмни елементи за конкретния етап от изпълнението на функциите и задачите; 3) Определяне на най-подходящия за изпълнение $PE_i$ в съответствие с избран критерий при изпълнение на съответен алгоритъм на управление $PRB_{i,j}^{solv}$; 4) Изследване и анализ на характеристиките на дадена структура на CPS при зададен дискретно разпределен граф на програмните елементи, изпълняващи функцията и задачата, както и при зададен алгоритъм на управление $MNG_k$; 5) Конкретни оценки и изводи. (11) Критериите за определяне на най-подходящ програмен елемент измежду всички свободни и изпълними програмни елемента са твърде много. Едни от най-важните са: 1) Минимален, максимален или случаен номер на програмния елемент, когато е номериран; 2) Минимално или максимално време за обработка на програмния елемент; 3) Минимален или максимален брой на разклоненията, съответно на междинните програмни връзки, които следват след изпълнението на програмния елемент. (12) Алгоритъмът на управление е най-добър, съответно времената за чакане на опашки в структурата са най-малки, когато алгоритъмът осигурява паралелност и независимост на изчислителния процес, а това става когато се избере програмен елемент с максимален брой разклонения и с най-малък пореден номер. Определяне на изследваните структури. В съответствие с направените в гл.2 аналитични изследвания и литературните проучвания [139, 144, 149, 150, 151] като най-подходящи са избрани девет структури, които за прегледност са групирани като матрица. Отчитайки изискванията на симулационните модели, като пряк обект за моделиране ще изберем структурите S11, S22 и S33, които са разположени по главния диагонал. Трябва да имаме предвид, че резултатите, валидни за тях, ще могат да се използват и за останалите структури със задоволително приближение. Избраните структури се характеризират със следните особености: 1) Сложността на структурата, като състав и организация нараства възходящо от S11 към S33; 2) Обект на изследване в структурите е функционалната специализация на процесорите при изменение на техния брой и начина на организация на връзките процесор-памет при изменение броя на процесорите, като за критерий се използва времето за обработка на програмен елемент; 3) Зависимостта на структурата по отношение на ресурси и управление от управляващите server/host/mainframe/domain системи намалява от S11 към S33; 4) Количеството на подчинената входно-изходна периферия се увеличава от S11 към S33; 5) Структурата S11 се използва в стандартни CPS (със средна цена), структурата S22 се използва в професионални CPS (с висока цена), а структурата S33 се използва в специални CPS (с много висока цена). Има изключения от това разделение, както и съответно смесено използване; 6) Избраните структури са идеализирани в разумни граници с цел намаляване сложността на симулационните модели; 7) За целите на симулационния анализ и за по-лесно отразяване динамиката на процесите [158, 159, 171, 190, 196], структурите ще включват в себе си повече от един процесор във всеки функционален тип.

Описание на структура с универсален процесор (S11). 1) Процесор - CPS е изградена върху структура, в която има само един тип процесор, които е универсален процесор. Обикновено това е 32 или 64 разряден микропроцесор, обработващ смесена (служебна/защитна и потребителска) информация при висока зависимост от управляващите server/host/mainframe/domain системи; 2) Памет: 1. Програмна памет - в нея е съсредоточено управлението на CPS. Ако е необходимо чрез специална процедура се извършва зареждане от управляващите server/host/mainframe/domain системи; 2. Даннова памет - в нея се зареждат входни данни, касаещи изпълнението на програмния елемент (също от управляващите server/host/mainframe/domain системи или от локални входно-изходни устройства). Освен това в данновата памет има области, имащи пряка връзка (извън управляващото устройство памет) с процесора. Тези области играят ролята на неявна работна памет, в която се записват междинните програмни връзки; 3) Локална входно/изходна периферия – тя е свързана към шинната периферия чрез управляващото устройство на периферията; 4) Линията – тя свързва ресурсите на CPS с управляващите server/host/mainframe/domain системи чрез управляващо устройство на линията. Етапите на обработка на $PE_j$ в S11 са следните: 1) SRO се изпълнява от входната периферия и от шинната периферия; 2) IRO се изпълнява от неявната работна памет (разположена в данновата памет) и шинната периферия, като конкретен $PRC_j^{uni}$ извършва четене от своя област за време $t^{iro}=0$. При четене от чужда област това време е различно от нула. То се извършва с помощта на управляващото устройство на паметта; 3) PRO се изпълнява от $PRC_k$ ; 4) IWO се изпълнява от данновата памет (областите на неявна работна памет) и шинната периферия, като записът се извършва само в областите, които са "свои" за даден процесор $PRC_k^{uni}$ ; 5) FWO се извършва от изходната периферия. Вижда се, че има конкуренция между SRO, FWO и IRO за шинната периферия. Проблемите се решават чрез приоритети (първите две ще имат приоритет 2, последната - приоритет 1) и подреждане на заявките за обслужване в опашки с дисциплина на обслужване FIFO. Най-сложният случай за тази структура е, когато $PRC_k^{uni}$ трябва да извърши IRO от "чужда", не своя област. Тогава $PRC_k^{uni}$ изпраща заявка до шинната периферия и управляващото устройство на паметта, и ако те са в състояние да обслужват, става прехвърляне на междинните резултати $b_{j,i}$ от областта на $PRC_j^{uni}$ чрез шинната периферия към областта от данновата памет, която принадлежи на $PRC_k^{uni}$ за някакво време $t^{iro}$. В случая е налице още едно ограничение: прехвърлянето на $b_{j,i}$ е възможно само, когато $PRC_j^{uni}$ извършва PRO. За да могат да се видят тесните места в работата на изследваните структури и съответно да ги сравним, ще разгледаме тяхната работа по изпълнение на един и същ примерен програмен елемент $PE_i$ , за който е необходимо да се извършат следните операции: 1) Да се извърши SRO за време $t^{sro}=t_1$ ; 2) Да се извърши IRO, при която да се осъществяват осем от десет възможни междинни програмни връзки за следните времена: 1. Междинна програмна връзка 1 с $PRC_j^{uni}$ за време $t^{iro1}=t_2$ ; 2. Междинна програмна връзка 2 със собствената област на данновата памет на $PRC_k^{uni}$ за време $t^{iro2}=t_3$ ; 3. Междинна програмна връзка 3 с $PRC_j^{uni}$ за време $t^{iro3}=t_4$ ; 4. Междинна програмна връзка 4 със собствената област на данновата памет на $PRC_k^{uni}$ за време $t^{iro4}=t_5$ ; 5. Междинна програмна връзка 5 с $PRC_j^{uni}$ за време $t^{iro5}=t_6$ ; 6. Междинна програмна връзка 6 – не се осъществява за време $t^{iro6}=t_7$ ; 7. Междинна програмна връзка 7 със собствената област на данновата памет на $PRC_k^{uni}$ за време $t^{iro7}=t_8$ ; 8. Междинна програмна връзка 8 с $PRC_j^{uni}$ за време $t^{iro8}=t_9$ ; 9. Междинна програмна връзка 9 – не се осъществява за време $t^{iro9}=t_{10}$ ; 10. Междинна програмна връзка 10 с $PRC_j^{uni}$ за време $t^{iro10}=t_{11}$ . 3) Да се извърши PRO за време $t^{pro}=t_{12}$ , като за това време $PRC_k^{uni}$ да прекъсне работата на два пъти за извършване на IRO от $PRC_j^{uni}$ за времена $t_{13}$ и $t_{14}$ ; 4) Да се извърши IWO, като се запише междинният резултат $b_{j,i}$ в собствената за $PRC_k^{uni}$ област от данновата памет за време $t^{iwo}=t_{15}$ ; 5) Да се извърши FWO на крайните резултати от изпълнението на $PE_i$ в изходната периферия или данновата памет за време $t^{fwo}=t_{16}$ . Тъй като извършването на междинните програмни връзки със собствената област на данновата памет за даден процесор ще става за време, което ще считаме за нула, поради пряката връзка между тях, то времената $t_3$ , $t_5$ , $t_8$ и $t_{15}$ са равни на нула. Освен тях, времената $t_7$ и $t_{10}$ са също равни на нула, тъй

като няма междинни програмни връзки. Следователно за $t_{S11}^{iro}$ и $t_{S11}^{cpo}$ можем да запишем: $t_{S11}^{iro}=t_2+t_4+t_6+t_9+t_{11}+t_{13}+t_{14}$ (3.22a) $t_{S11}^{cpo}=t_{12}+t_{13}+t_{14}$ (3.22b). Изводът, който може да се направи от (3.22a и 3.22b) е, че за да се намалят тези най-важни за обработката времена, трябва да се ограничат междинните програмни връзки и прекъсванията на процесорите. По-нататъшната оптимизация на структурата трябва да се извършва по линията на организацията на паметта и функционалната спецификация на процесорите, което ще премахне прекъсванията. Това е направено в следващата изследвана структура.

Описание на структура със защитен копроцесор (S22). 1) Процесор - в този случай, структурата върху която е изградена CPS се отличава с това, че специализираната (служебна/защитна) информация се обработва отделно от останалата потребителска информация от специализиран защитен процесор ($PRC^{pro}$). Зависимостта от управляващите server/host/mainframe/domain системи е средна и се използва служебна/защитна процедура с макроезик. Структурата има два типа процесори: $PRC^{uni}$ и $PRC^{pro}$. Те са обикновено 32 или 64 разрядни микропроцесори от конвенционален или специализиран тип; 2) Памет: 1. Програмна памет – със същите функции както при S11; 2. Даннова памет – също както при S11, с тази разлика, че няма области на неявна работна памет; 3. Работна памет – паметта, в която се записват междинните резултати $b_{j,i}$ при изпълнението на програмен елемент с цел реализиране на междинна програмна връзка. 3) Управляващите устройства за линията и периферията са същите както при S11, за разлика от управляващото устройство на паметта, което осъществява цялостното управление на връзките процесор - памет без наличието на пряко свързване; 4) Входната и изходната периферия са същите с някои допълнения, улесняващи операторската работа; 5) Шинната периферия съдържа две самостоятелни шинни системи: външна ($PRF_{exter}^{bus}$), по която се извършват началната четяща операция и крайната записваща операция, и вътрешна ($PRF_{inter}^{bus}$), по която се извършват междинната четяща операция и междинната записваща операция. Етапите на обработката на $PE_j$ в S22 са също пет, както в S11, но имат разлика в обвързването на ресурсите с отделните операции с цел премахване или намаляване на тесните места: 1) Началната четяща операция се изпълнява от входната периферия чрез $PRF_{exter}^{bus}$ и от данновата памет отново чрез $PRF_{exter}^{bus}$; 2) Междинната четяща операция се изпълнява от работната памет чрез $PRF_{inter}^{bus}$, като управляващото устройство на паметта управлява изпълнението на заявките на процесорите за извършване на IRO за време $t^{iro}$, винаги отлично от нула за разлика от S11; 2) Обработващата операция се изпълнява от $PRC^{uni}$ и $PRC^{pro}$, като те работят паралелно във времето и, позовавайки се на [209, 213, 223, 243] ще приемем, че в даден момент от време обработващата операция е разделена между двата процесора в съотношение съответно 1:2; 3) Междинната записваща операция се изпълнява от работната памет чрез $PRF_{exter}^{bus}$ под управлението на управляващото устройство на паметта; 4) Крайната записваща операция се извършва от изходната периферия чрез $PRF_{inter}^{bus}$. При изпълнението на примерния $PE_1$ от S22 се вижда, че в сравнение със S11 само времената $t_7$ и $t_{10}$ са равни на нула, тъй като в примера няма междинни програмни връзки. Следователно за $t_{S22}^{iro}$ и $t_{S22}^{cpo}$ можем да запишем: $t_{S22}^{iro}=t_2+t_3+t_4+t_5+t_6+t_8+t_9+t_{11}$ (3.23a) $t_{S22}^{cpo}=t_{12}+t_{15}$ (3.23b). Изводът, който може да се направи от (3.23a) и (3.23b) е, че в сравнение със S11 прекъсванията са премахнати, но са увеличени времената за междинните четящи операции като количество поради това, че е премахната пряката връзка между процесора и паметта. Освен това, времето на обработващата операция е рязко намалено за сметка на функционалната специализация на процесорите и паралелната им работа. По-нататъшната оптимизация на структурата трябва да се извършва по линията на намаляване на $t^{iro}$ чрез осигуряване на пряка връзка между процесора и паметта (като всеки процесор си има своя работна и даннова памет), както и чрез намаляване на $t^{cpo}$ чрез по-голяма специализация на процесорите при запазване на паралелната им работа. Това е направено в следващата структура.

Описание на структура със защитни копроцесори (S33). 1) Процесор - CPS, изградена на основата на тази структура, има ниска зависимост от управляващите server/host/mainframe/domain системи и се използва преди всичко в специализирани случаи от висок клас. Обработва служебна/защитна информация с много висока динамичност в реално време заедно с потребителския информационен поток. Осигурено е наличието на богат избор от хардуерни и софтуерни средства, формиращи ресурсите

на структурата. Както вече беше споменато, броят на копроцесорите може да бъде над 20, но ние ще се ограничим да разгледаме структурата само с четири копроцесора, изхождайки от целите за оптималност на симулационните модели и от целите на настоящата дисертация. Четирите типа копроцесори са: $PRC^{tcp}$, $PRC^{ip}$, $PRC^{udp}$ и $PRC^{icmp}$. За да се осигури необходимата работоспособност и бързодействие, е добре процесорите да бъдат 32 или 64 разрядни универсални или специализирани микропроцесори; 2) Памет: 1. Програмна памет – в нея е съсредоточено цялото управление на структурата, съдържащо една или повече операционни системи, специални системни средства за работа с езици от високо ниво, асемблери, моделиращи програми, приложни програми и други; 2. Даннова памет – съдържа входните данни, които подлежат на обработка в съответствие с приложните програми. Тя е разделена на четири групи от независими области, като всяка група съдържа толкова области, колкото процесори влизат в състава на функционалната група. Общият брой области в четирите групи трябва да бъде равен на общия брой процесори в четирите функционални типа. Всеки от процесорите има пряка връзка със своята област. От нея може да се осъществява SRO, FWO и непряка връзка (чрез управляващото устройство на паметта) с чуждите области, от и в които може да се осъществяват същите операции; 3. Работна памет – съдържа междинните резултати от изчислителния процес, при което тя е организирана по същия начин (четири групи от области и т.н.) както данновата памет. Аналогично, всеки от процесорите има пряка връзка със своята област, от и в която може да осъществява IRO, IWO и непряка връзка (чрез управляващото устройство на паметта) с чуждите области, от и в които може да реализира същите операции. В по-нататъшното разглеждане ще използваме означението $AMEM_k^{x,yyy}$, което ще означава област от паметта x (даннова или работна), принадлежаща на k-тия процесор от yyy-тия функционален тип (tcp, ip, udp и icmp). 3) Управляващото устройство на линията и на периферията имат същите задачи, както и в останалите структури, а управляващото устройство на паметта осигурява запис и четене на информация за всички процесори в области, които са чужди за тях. Възможно е трите управляващи устройства да бъдат заменени съответно със специализирани процесори ($PRC^{lin}$, $PRC^{prf}$ и $PRC^{mem}$), но в нашия случай, за да не усложняваме моделите, ще приемем, че това са само управляващи устройства; 4) Входната и изходната периферия имат широко представителство и облекчават работата на оператора, като позволяват и дублиране на устройствата; 5) Шинната периферия съдържа същите две самостоятелни шинни системи, както и S22. Етапите на обработка на $PE_i$ в S33, подобно на другите структури са също пет, като е налице разлика само в обвързването на ресурсите с изпълнението на операциите, чрез което се намаляват тесните места в структурата: 1) SRO се изпълнява от входната периферия, $MEM^{d,tcp}$, $MEM^{d,ip}$, $MEM^{d,udp}$ и $MEM^{d,icmp}$ чрез $PRF_{inter}^{bus}$; 2) IRO се изпълнява от $MEM^{w,tcp}$, $MEM^{w,ip}$, $MEM^{w,udp}$ и $MEM^{w,icmp}$ чрез $PRF_{exter}^{bus}$, като $t^{iro}$ е равно на нула за свои области на даден процесор и различно от нула за чужди области (заявките за обслужване се управляват от управляващото устройство на паметта); 3) PRO се изпълнява от $PRC^{tcp}$, $PRC^{ip}$, $PRC^{udp}$ и $PRC^{icmp}$, които работят паралелно във времето и, позовавайки се на [245, 249, 253, 260], ще приемем, че в даден момент от време обработващата операция е разделена между четирите типа процесори в съотношение съответно (3:3:2:2); 4) IWO се изпълнява от $MEM^{w,tcp}$, $MEM^{w,ip}$, $MEM^{w,udp}$ и $MEM^{w,icmp}$ в областите, които са свои за даден тип процесор чрез $PRF_{inter}^{bus}$. В областите, които са чужди за даден тип процесор, тя се изпълнява под управление на управляващото устройство на паметта също чрез $PRF_{inter}^{bus}$; 5) FWO се извършва от изходната периферия, $MEM^{d,tcp}$, $MEM^{d,ip}$, $MEM^{d,udp}$ и $MEM^{d,icmp}$ чрез $PRF_{exter}^{bus}$. От изпълнението на примерния $PE_i$ се вижда, че в сравнение със S22 времената $t_3$, $t_5$, $t_7$, $t_8$, $t_{10}$ и $t_{15}$ са равни на нула, поради преките връзки процесор - памет. Следователно: $t_{S33}^{iro}=t_2+t_4+t_6+t_9+t_{11}$ (3.24а) $t_{S33}^{epo}=t_{12}$ (3.24b). Изводът, който може да се направи от (3.24а) и (3.24b) е, че в сравнение със S11 прекъсванията са премахнати, а в сравнение със S22 са ограничени до минимум времената за IRO като количество, поради наличието на преки връзки процесор - памет и още повече е намалено времето за обработващата операция за сметка на функционалната специализация на процесорите и паралелната им работа. На този етап оптимизацията на структурата на CPS е реализирана в първо приближение. По-нататъшната задача е определяне на оптималния брой процесори, влизащи в състава на всеки функционален тип при определени други ресурси.

*Избор на метод за изследване (SM1-2).* Разгледаните по-горе проблеми, свързани с определянето на типа на FT', с определянето на ресурсите и тяхното взаимодействие, и с определянето на вариантите на структурите, върху които се изгражда CPS, недвусмислено ни убеждават, че един от най-добрите методи за изследване на една толкова сложна система с голям брой динамично изменящи се и взаимно влияещи си параметри, с ярко изявен стохастичен характер на протичане на процесите в нея, с различни приоритетни дисциплини на обслужване на многобройни еднородни и разнородни опашки е симулационното моделиране с използване на т.нар. опашкови мрежи, известни още и под името теория на масовото обслужване, като се реализират компютърни експерименти върху симулираща изчислителна система.

*Прогноза за възможни решения (SM1-3).* Тази стъпка присъствува в неявен вид по време на цялата дейност в стъпка SM1-1, тъй като определянето на типа на FT', на състава, структурата и управлението на CPS е свързано с прогнозиране на възможните решения на конкретните задачи. Сега остава да се огледат още веднъж направените предположения за ресурсите и взаимодействието им, но като се държи сметка за изискванията към базата на изследване и ресурсите на симулиращата изчислителна система. Трябва да се прецени, дали сложността на определените ресурси на CPS позволява поставените задачи за симулационно изследване да бъдат решени с възможните и налични симулиращи ресурси. Изводът, който трябва да се направи е, че идеализирането и детайлността в описанието на ресурсите и взаимодействието помежду им са направени в разумни граници, съобразени с възможностите на наличната симулираща система.

*Избор на база за изследване (SM1-4).* Следвайки всички направени досега оценки и изводи, можем да изберем като база за изследванията симулираща система от типа PC Base в комбинация с общоцелевата симулационна програма GPSS/H под управлението на операционна система MS Windows 98.

*Определяне на ресурси за симулиращата система (SM1-5).* Минималните изисквания към ресурсите на симулиращата система включват две процесорни конфигурации с Intel Pentium III – 600 и оперативна памет 128 MB за всяка, две дискови устройства с размер 15 GB, две CD-RW и комуникационна линия за Интернет операции с гарантирана лента около 28,8 Kbits/sec; основна операционна система със съответните мрежови драйвери за периферията от тип MS Windows 98 и съответна MS Windows версия на GPSS/H. При тези условия необходимото компютърно време за експерименти със симулационните модели на CPS е около 960 часа (без конкретно планиране).

ФОРМИРАНЕ НА ОБЩ СИМУЛАЦИОНЕН МОДЕЛ. Поради съображенията за краткост стъпките в етапи SM2 и SM3 няма да бъдат разглеждани самостоятелно, а ще преминем към стъпките на етап SM4 – формиране на общ модел, където те ще присъствуват в неявен вид. По същите причини няма да разглеждаме самостоятелно, стъпките на оптимизационния процес от етапи OP1, OP2, OP3, OP4 и OP5, но те ще присъствуват в неявен вид при съответните разглеждания. .

*Създаване на концептуален модел (SM4-20).* Създаването на концептуален модел на CPS ще се основава на постановките, изложени по-горе, и на разбирането, че той ще бъде резултат от взаимодействието на формалните модели на съставните елементи на системата. Те обединяват твърде разнородна и сложна съвкупност от компоненти. Създаването на коректен и работоспособен симулационен модел на защитната система при тези начални условия е теоретически възможно, но практически неосъществимо. Ето защо е необходимо да се направят необходимите ограничения и опростявания с цел получаване на използваеми резултати от симулационното изследване. Конкретно това ще се изрази в следното: 1) Създаване на формален модел за избран тип функции и задачи, чийто алгоритъм позволява представянето му чрез дискретно разпределен граф; 2) Създаване на формални модели за избраните структури, по такъв начин, че да бъдат обединени в едно останалите съставни елементи.

Формален модел на избран тип задачи. Моделът ще бъде реализиран въз основа на предположението, че алгоритъмът на избрания тип функции и задачи може да се представи чрез дискретно разпределен граф, който се описва от следната четворка: $DDG=(InD,OuD,PE,IPB)$ (3.25) където: $InD^{FT}=(ind_{a,p}^{FT})_{a,p}^{\in A,P}$ е множеството на входните данни на функциите и задачите; $OuD^{FT}=(oud_{f,m}^{FT})_{f,m}^{\in F,M}$ е множеството на изходните резултати; $PE^{FT}=(pe_i^{FT})_i^{\in I}$ е множеството на програмните елементи, на които е разделена функцията и задачата; $IPB=(b_{i,j})$ е множеството на междинните програмни връзки, които ще осъществят програмните елементи. Отчитайки стохастичния характер на процесите, параметрите на конкретен дискретно разпределен граф могат да се опишат чрез следните функции на разпределение на

вероятности (FDP): 1) $F^{01}$ – функция на времето за изпълнение на началната четяща операция; 2) $F^{02}$ – функция на времето за изпълнение на крайната записваща операция; 3) $F^{03}$ – функция на времето за изпълнение на една операция в един програмен елемент; 4) $F^{04}$ – функция на броя на нивата, през които се осъществява междинната програмна връзка в двете посоки; 5) $F^{05}$ – функция на броя на програмните връзки, влизащи / излизащи в един програмен елемент; 6) $F^{06}$ – функция на броя на програмните елементи в отделно ниво; 7) $F^{*}$ – функция на броя операции в програмен елемент; 8) $F^{*}$ – функция на времето за изпълнение на IRO; 9) $F^{**}$ - функция на броя на началните и крайните операции; 10) $F^{10}$ - функция на броя на нивата за дискретно разпределен граф; 11) $F^{11}$ - функция на времето за изпълнение на обработващата операция; 12) $F^{12}$ - функция на времето за изпълнение на междинната записваща операция. Съвкупността от горните функции описва в достатъчна степен голям брой варианти на дискретно разпределен граф, представящи приемлив брой варианти на избрания тип функции и задачи.

Формален модел на избрани структури. Той ще бъде изграден със средствата на опашковите мрежови модели. За тази цел трябва да разглеждаме процесите в ресурсите на структурите и тяхното взаимодействие като процеси на издаване на запитвания и тяхното обслужване от обслужващи прибори, като тези процеси са недетерминирани и стохастични. Формалните модели на S11, S22 и S33 са създадени с помощта на многоканални прекъснати отворени стохастични опашкови мрежи. Съществува възможността за външен източник на задания, например управляващите server/host/mainframe/domain системи, които имат специално отношение към ресурса управление. Характерът на потоците от запитвания за обслужване е стохастичен, като времената за чакане пред обслужващия прибор се представят като времена за чакане на опашки и накрая във формалните модели има повече от един обслужващ прибор (поради това се наричат мрежи). Като цяло, всяка опашкова мрежа е разделена на три подсистеми: 1) Подсистема "Управление-Процесор", в която са представени взаимоотношенията между ресурсите управление и процесор. С помощта на QUE1 са представени запитванията към управлението за получаване на задание (програмен елемент) за обслужване от страна на процесорите. В случай на функционална специализация на процесорите (S22 и S33) QUE1 се разделя на няколко вторични опашки пред всеки функционален тип процесор: $QUE1^{um}$ и $QUE1^{pm}$ за S22; $QUE1^{sp}$, $QUE1^{ip}$, $QUE1^{udp}$ и $QUE1^{icmp}$ за S33. От своя страна, всяка вторична опашка се разделя на q третични опашки, където q е броят на процесорите във всеки функционален тип; 2) Подсистема "Шинна Периферия – Входна Периферия / Изходна Периферия", в която са представени взаимоотношенията между отделните части на ресурса периферия. С помощта на QUE2 са представени запитванията към периферията за извършване на съответна операция в изчислителния процес. При S22 и S33 шинната периферия се разделя на два подресурса $PRF^{bus}_{exter}$ и $PRF^{bus}_{inter}$, а това означава, че QUE2 ще се раздели на две вторични опашки $QUE2^{exter}$ и $QUE2^{inter}$; 3) Подсистема "Даннова Памет – Работна Памет", в която са представени взаимоотношенията между отделните части на ресурса памет. С помощта на QUE3 са представени запитванията към паметта за извършване на съответна операция. При S22 и S33 паметта се разделя на два подресурса даннова памет и работна памет, а това означава, че QUE3 ще се раздели на две вторични опашки $QUE3^{data}$ и $QUE3^{work}$. Освен това, в S33 данновата и работната памет стават многоканални, а това означава, че всяка вторична опашка се разделя на k третични опашки, където k е броят на областите от паметта (AMEM) за всеки функционален тип процесор. Гореописаните формални модели на избраните структури на CPS заедно с формалните модели на избрания тип функции и задачи чрез своето взаимодействие ще образуват концептуалния модел за цялата защитна система, като за целите на нашето изследване ще разгледаме неговото поведение в динамичен режим. Това ще направим, като разгледаме функционирането на структурите S11, S22 и S33 при изпълнение на програмен елемент в условията на опашковите мрежи. По такъв начин, към времената за изпълнение на отделните операции ще прибавим и конкретни времена на програмния елемент $PE_i$ в различните опашки пред обслужващите прибори. В израза (3.20) във времето $t^{pe}_{i,k}$ е включено и времето $t^{wait}_{i,k}$, което сега ще бъде разложено на своите съставящи, като за удобство ще използваме изпълнението на примерния $PE_i$. За структурата S11 ще бъде в сила следното равенство обединяващо сумираните времена:

$t_{i,k}^{pe} = \mathbf{t}_{que2}^{sro} + t_{i,k}^{sro} + \mathbf{t}_{que2}^{iro} + t_{i,k}^{iro1} + t_{i,k}^{iro3} + \mathbf{t}_{i+1,j}^{irq1,iro} + t_{i,k}^{iro5} + t_{i,k}^{iro8} + \mathbf{t}_{i+1,j}^{irq2,iro} + t_{i,k}^{iro10} + t_{i,k}^{pro'} + \mathbf{t}_{i+1,j}^{irq1,pro} + t_{i,k}^{pro''} + \mathbf{t}_{i+1,j}^{irq2,pro} + t_{i,k}^{pro'''} + \mathbf{t}_{que2}^{fwo} + t_{i,k}^{fwo}$ (3.26) където: $\mathbf{t}_{que2}^{sro}$ е времето за чакане на запитване за SRO на $PRC_k^{uni}$ пред *QUE2*; $\mathbf{t}_{que2}^{iro}$ е времето за чакане на запитване за IRO на $PRC_k^{uni}$ пред QUE2; $\mathbf{t}_{i+1,j}^{irq1,iro}$ е времето за чакане при първото прекъсване на $PRC_k^{uni}$ по време на IRO от страна на $PRC_j^{uni}$, извършващ SRO или FWO на $PE_{j-1}$; $\mathbf{t}_{i+1,j}^{irq2,iro}$ е времето за чакане при второто прекъсване и е аналогично на предходното; $\mathbf{t}_{i+1,j}^{irq1,pro}$ е времето за чакане при първото прекъсване на $PRC_k^{uni}$ по време на CPO от страна на $PRC_j^{uni}$, извършващ IRO за $PE_{i-1}$; $\mathbf{t}_{i+1,j}^{irq2,pro}$ е времето за чакане при второто прекъсване и е аналогично на предходното; $\mathbf{t}_{que2}^{fwo}$ е времето за чакане на запитване за FWO на $PRC_j^{uni}$; $t_{que2}^{sro} = t_1$; $t_{i,k}^{iro1} = t_2$; $t_{i,k}^{iro3} = t_4$; $t_{i,k}^{iro5} = t_6$; $t_{i,k}^{iro8} = t_9$; $t_{i,k}^{iro10} = t_{11}$; $t_{i,k}^{pro'} = t_{12'}$; $t_{i,k}^{pro''} = t_{12''}$; $t_{i,k}^{pro'''} = t_{12'''}$; $t_{i,k}^{fwo} = t_{16}$. За структурата S22 ще бъде в сила следното равенство, обединяващо сумираните времена, където са отразени нейните специфични особености в сравнение с първата структура: $t_{i,k}^{pe} = \mathbf{t}_{que2^m}^{sro} + t_{i,k}^{sro} + \mathbf{t}_{que3^w}^{iro} + t_{i,k}^{iro1} + t_{i,k}^{iro2} + t_{i,k}^{iro3} + t_{i,k}^{iro4} + t_{i,k}^{iro5} + t_{i,k}^{iro7} + t_{i,k}^{iro8} + t_{i,k}^{iro10} + K^{fpu} t_{i,k}^{pro} + \mathbf{t}_{que3^w}^{iwo} + t_{i,k}^{iwo} + \mathbf{t}_{que2^{ex}}^{fwo} + t_{i,k}^{fwo}$ (3.27) където: $\mathbf{t}_{que2^m}^{sro}$ е времето за чакане на запитване на SRO на $PRC_k^{sxx}$ пред $QUE2^m$; $\mathbf{t}_{que3^w}^{iro}$ е времето за чакане на запитване за IRO на $PRC_k^{sxx}$ пред $QUE3^w$; $K^{fpu}$ е коефициент на функционална процесорна универсалност (за S22 е 0,6 [7, 12]), отчитащ намалението на $t_{i,k}^{pro}$ поради функционалната специализация на процесорите и паралелната им работа: $\mathbf{t}_{que3^w}^{iwo}$ е времето за чакане на запитване за IWO на $PRC_k^{sxx}$ пред $QUE3^w$; $\mathbf{t}_{que2^{ex}}^{fwo}$ е времето за чакане на запитване на FWO на $PRC_k^{sxx}$ пред $QUE2^{ex}$; $t_{i,k}^{sro} = t_1$; $t_{i,k}^{iro1,2,3,4,5,7,8,10} = t_2, t_3, t_4, t_5, t_6, t_8, t_9, t_{11}$; $t_{i,k}^{pro} = t_{12}$; $t_{i,k}^{iwo} = t_{15}$; $t_{i,k}^{fwo} = t_{16}$. За структурата S22 ще бъде в сила следното равенство обединяващо сумираните времена, където са отразени нейните специфични особености в сравнение с първата структура: $t_{i,k}^{pe} = \mathbf{t}_{que2^m}^{sro} + t_{i,k}^{sro} + \mathbf{t}_{que3_j^{m,sxx}}^{iro1} + t_{i,k}^{iro1} + \mathbf{t}_{que3_j^{m,sxx}}^{iro3} + t_{i,k}^{iro3} + \mathbf{t}_{que3_j^{m,sxx}}^{iro5} + t_{i,k}^{iro5} + \mathbf{t}_{que3_j^{m,sxx}}^{iro8} + t_{i,k}^{iro8} + \mathbf{t}_{que3_j^{m,sxx}}^{iro10} + t_{i,k}^{iro10} + K^{fpu} t_{i,k}^{pro} + \mathbf{t}_{que2^{ex}}^{fwo} + t_{i,k}^{fwo}$ (3.28) където: $\mathbf{t}_{que2^m}^{sro}$ е аналогично на (3.27); $\mathbf{t}_{que3_j^{m,sxx}}^{iro1,3,5,8,10}$ е времето за чакане на запитване за IRO 1, 3, 5, 8. 10 на $PRC_k^{sxx}$ пред $QUE_j^{m,sxx}$ на $AMEM_j^{m,sxx}$, която принадлежи на $PRC_j^{sxx}$ и е чужда за $PRC_k^{sxx}$; всички останали времена са аналогични на тези в (3.27), като $K^{fpu}$ има стойност 0,3 за S33 [7, 12]. В заключение на анализа на функционирането на формалните модели на избраните структури, изградени с помощта на опашковите мрежи, може да се направи изводът, че прибавените времена за чакане на опашки и за прекъсване на процесорите (всички времена, които не са с курсив) потвърждават възходящата оптималност на структурите (от S11 към S33) по отношение на времето за изпълнение на отделен програмен елемент или на цялата функция и задача.

*Определяне на критерии за оценка (SM4-21).* Това е стъпка с основно значение за симулационното изследване, защото проведените експерименти и анализът на резултатите ще се осмислят само при наличието на коректно зададени, функционално пълни и достатъчно обобщаващи критерии за оценка на симулационното изследване, които при наличие на задачи за оптимизация ще бъдат и критерии на оптимизационен процес, като всеки критерий на оптимизационен процес е едно обобщение на група от параметри на оптимизационен процес, представящи различни свойства на реалния обект за оптимизация.

Общи критерии за оценка. Това са критерии, които оценяват CPS преди всичко от потребителски аспект, като те са следните: 1) Системно време за обработка $TIM^{sys}$ – времето, за което е обработена дадена функция и задача, представена чрез своя дискретно разпределен граф. $TIM^{sys} = t^{work} + t^{wat}$ (3.29) където: $t^{work}$ е времето за работа; $t^{wat}$ е времето за чакане на опашки пред различните ресурси на защитната система, като ще се уговорим това да бъдат хардуерните ресурси, приемайки, че в програмните ресурси времето за чакане е нула, което влиза в рамките на допустимата грешка. Освен това $t^{work}$ е равно на: $t^{work} = t^{sro} + t^{iro} + t^{pro} + t^{fwo} = t_{useful}^{work} + t_{unuseful}^{work}$ (3.30), $t_{useful}^{work} = t^{sro} + t^{pro} + t^{fwo}$ (3.31), $t_{unuseful}^{work} = t^{iro} + t^{iwo}$ (3.32), където: $t_{useful}^{work}$ е полезното, а $t_{unuseful}^{work}$ е не полезното работно време. Тъй като всяка функция и задача представлява съвкупност от отделни програмни елементи, то времената в горните формули се явяват

обобщение за цялата функция и задача, обединявайки времената на отделните операции за всички програмни елементи. С цел опростяване на изследването ще предполагаме, че в рамките на една функция и задача времената $t^{src}$ и $t^{fmv}$ няма да се изменят, а оптимизирането на избраните структури ще се изразява в изменението на останалите съставящи, като $t^{pro}$ ще отразява степента на функционална специализация и паралелност в работата на процесорите, а $t^{tro}$ ще отразява оптималността на дискретно разпределен граф за всяка функция и задача към дадена структура и конфигурация на CPS; 2) Системна скорост на обработка $SPD^{sys}$ – това е броят на програмните елементи, изпълнявани за единица време, т.е.: $SPD^{sys}=\frac{C}{TIM^{sys}}$ (3.33), където C е броят на програмните елементи, влизащи в състава на отделната функция и задача. Този критерий оценява директно бързодействието на ресурсите; 3) Системна цена на обработка $PRI^{sys}$ – това е стойността на обработката на дадена функция и задача, изразена в предварително дефинирани парични единици. В нея се включват стойността на HT' и ST', както и евентуално таксата за наети под наем ресурси (например, комуникационна линия).

Специфични критерии за оценка. Това са критерии, които ще оценяват характеристиките на CPS от гледна точка на проектанта, като ще се отнасят за отделен ресурс (подсистема) или за системата като цяло. 1) Заетост LOD – това е отношение на времето на заетост на даден ресурс при изпълнение на дадена функция и задача $t^{load}$, към $TIM^{sys}$: $LOD=\frac{t^{load}}{TIM^{sys}}$ (3.34); 2) Брой обслужени запитвания NSI – показва броя на запитванията, обслужени от даден ресурс за времето, през което той е активен $t^{load}$; 3) Време за обслужване на запитване TSI – ще използваме средната стойност на този критерий, която е равна на отношението $TSI=\frac{t^{load}}{NSI}$ (3.35); 4) Продуктивност PRD – това е обемът информация, която се обработва за единица време в изследваната структура при изпълнение на дадена функция и задача : $PRD=\frac{SCO^{inf}}{TIM^{sys}}$ (3.36); 5) Системна относителна продуктивност $PRD^{sys}$ – изразява се чрез отношението на продуктивността на структура на защитна система, в която има еднотипни процесори на брой q, към продуктивността на структурата на защитна система, в която има многотипни процесори, като броят на типовете се изменя до $q^*$, с текущ индекс i, а броят на количеството процесори от всеки тип се изменя до q, с текущ индекс j, или $PRD^{sys}=\frac{PRD_q}{PRD_{q_i \cdot q_i}}=\frac{TIM^{sys}_{q_i \cdot q_i}}{TIM^{sys}_{q}}$ (3.37); 6) Системна средна продуктивност $PRD^{mid}$ – изразява се чрез отношението на системната относителна продуктивност към произведението от броя на процесорите в структурата и коефициента на функционална процесорна универсалност $K^{fpu}$ на дадена структура: $PRD^{mid}=\frac{PRD^{sys}}{qK^{fpu}}$ (3.38) където $K^{fpu}=\frac{1}{PSCO_1^{inf}+PSCO_2^{inf}+...+PSCO_{q^*}^{inf}}\frac{PSCO^{inf}_{max}}{}$ (3.39), като с $PSCO_{q^*}^{inf}$ са означени частите от общия обем информация, която се обработва от всеки тип копроцесор, а с $PSCO^{inf}_{max}$ е означена най-голямата от всички части. Съгласно [7, 12] за избраните S11, S22 и S33 стойността на $K^{fpu}$ е съответно 1; 0,6; 0,3. В зависимост от нуждите, може да се използва и реципрочният на $K^{fpu}$ коефициент на функционална процесорна специализация $K^{fps}$; 7) Производителност $PRF^{sys}$ – това е количеството изчислителна работа, извършена за единица време в изследваната структура при изпълнение на дадена функция и задача, или $PRF^{sys}=\frac{WRK^{comp}}{TIM^{sys}}$ (3.40) където $WRK^{comp}=\sum_{j}^{n1}\sum_{i}^{n2}NUI_{i,j}^{xxx}t_jC_{i,j}^{xxx}$ (3.41) и с $NUI_{i,j}^{xxx}$ е означен броят на инструкциите, изпълнявани от i-тия ресурс от тип xxx в j-тия интервал от време, с $t_j$ е означена продължителността на времевия интервал

j, а с $C_{i,j}^{xxx}$ е означена константа, показваща дали i-тия ресурс от тип xxx работи (стойност 1) или не работи (стойност 0) в интервала $t_j$ ; 8) Пълна производителност $PRF^{tot}$ – изразява се с пълната $WRK^{comp}$, извършвана от всички ресурси на структурата при изпълнение на дадена функция и задача, или $PRF^{tot}=\frac{WRK_{prc}^{comp}+WRK_{mem}^{comp}+WRK_{prf}^{comp}}{TIM^{sys}}$ (3.42) а освен това $PRF^{tot}=PRF^{mem}+PRF^{prf}+PRF^{prc}=\sum_{i=1}^{k}PRF_i^{mem}+\sum_{i=1}^{l}PRF_i^{prf}+\sum_{i=1}^{q}PRF_i^{prc}$ (3.43), където: с k, l и q са означени съответните количества от всеки тип ресурс, влизащи в състава на структурата на защитната система, при което $TIM^{sys}$ включва в себе си само времена за полезна SRO, PRO и FWO и неполезна IRO и IWO работа, като времената за чакане на опашки са изключени; 9) Полезна производителност $PRF_{useful}^{xxx}$ – характеризира производителността на процесорната подсистема (в този случай) на структурата, като се включват само времената за извършване на полезни операции, или $PRF_{useful}^{prc}=\frac{WRK_{useful}^{comp}}{TIM^{sys}}=WRK_{sro}^{comp}+WRK_{pro}^{comp}+\frac{WRK_{fwo}^{comp}}{TIM^{sys}}$ (3.44); 10) Реално полезна производителност $PRF_{r\text{-}useful}^{xxx}$ – характеризира производителността на процесорната подсистема (в този случай), но отчита и случаите, когато има съвместяване по време на началната четяща и крайната записваща с обработващата операция, при което полезната процесорна производителност се увеличава, или $PRF_{r\text{-}useful}^{prc}=\frac{WRK_{prc}^{comp}-WRK_{prf}^{comp}}{TIM^{sys}}$ (3.45); 11) Специфично полезна производителност $PRF_{s\text{-}useful}^{xxx}$ – характеризира производителността на процесорната подсистема (в този случай), но отчита и случаите, при които има съвместяване по време на началната четяща и крайната записваща с обработващата операция, при което полезната процесорна производителност се увеличава, или $PRF_{s\text{-}useful}^{prc}=\frac{WRK_{prc}^{comp}+WRK_{prf}^{comp}}{TIM^{sys}}$ (3.46); 12) Системна използваемост $USB^{sys}$ – това е отношението на изчислителната работа, извършвана от даден ресурс или от цялата система, към максимално възможната за съответния ресурс или съответната система от ресурси, или $USB^{sys}=\frac{WRK_{xxx,current}^{comp}}{WRK_{xxx,max}^{comp}}$ (3.47) където с $WRK_{xxx,cur}^{comp}$ е означена текущата стойност на работата, извършвана от даден ресурс, като с xxx е означено, че съществуват различни видове изчислителни работи (с различни съставящи); 13) Полезна използваемост $USB_{useful}^{xxx}$ – отразява полезната използваемост на процесорната подсистема (в този случай) и по нейната стойност може да се съди, доколко е ефективна дадена структура на защитната система, или $USB_{useful}^{prc}=\frac{WRK_{useful}^{comp}}{PRF_{useful}^{prc}TIM^{sys}}$ (3.48); 14) Реално полезна използваемост $USB_{r\text{-}useful}^{xxx}$ – отразява полезната използваемост на процесорната подсистема (в този случай), но като отчита и случаите, при които няма съвместяване по време на началната четяща и крайната записваща с обработващата операция, в резултат на което има намаляване на полезната процесорна използваемост, или $USB_{r\text{-}useful}^{prc}=\frac{WRK_{r\text{-}useful}^{comp}}{PRF_{r\text{-}useful}^{prc}TIM^{sys}}$ (3.49); (15) Специфична полезна използваемост $USB_{s\text{-}useful}^{xxx}$ – отразява полезната използваемост на процесорната подсистема (в този случай), но като отчита случаите, при които има съвместяване по време на началната четяща и крайната записваща с обработващата операция, в резултат на което има увеличаване на полезната процесорна използваемост, или $USB_{s\text{-}useful}^{prc}=\frac{WRK_{s\text{-}useful}^{prc}}{PRF_{s\text{-}useful}^{prc}TIM^{sys}}$ (3.50); 16) Пълна използваемост $USB^{tot}$ – представлява обобщение на използваемостта на отделните ресурси и подсистеми, по отношение на цялата структура на защитната система, или $USB^{tot}=USB^{mem}+USB^{prf}+USB^{prc}=\sum_{i=1}^{k}USB_i^{mem}+\sum_{i=1}^{l}USB_i^{prf}+\sum_{i=1}^{q}USB_{i,i}^{prc}$ (3.51), а освен

това $USB^{tot} = \frac{WRK_{tot}^{comp}}{PRF^{tot} TIM^{sys}}$ (3.52). В заключение на стъпката (SM4-21) за определяне на критериите за оценка на симулационното изследване, ще се уговорим, че с цел опростяване на записването в горните изрази (с изключение на критерий 6) е изпуснат $K^{fpu}$, който ще бъде въвеждан в по-нататъшните разглеждания когато е нужно; освен това някои от критериите (9, 10, 11, 13, 14 и 15) са разгледани за процесорната подсистема като най-характерен случай в съответствие с поставените пред дисертацията задачи, но те са валидни и за другите подсистеми, което ще бъде отбелязано по-нататък, когато е необходимо. Освен това, при определяне на критериите за оценка като основа са използвани идеите и резултатите в [261, 262, 265, 269, 273, 279, 288].

*Определяне на съотношения между критерии (SM4-22).* За да могат да бъдат описани свойствата и поведението на CPS като обект на симулационно изследване, критериите за оценка трябва да бъдат обвързани чрез математически зависимости, в които да се включат определено количество параметри, представящи определено количество свойства на защитната система, като всеки един от параметрите може да бъде константа или функция и заедно с това, когато има оптимизационен процес, всеки един от параметрите може да е параметър на този процес. В съответствие с направените разглеждания по-горе ще групираме по нов начин типовете параметри на оптимизационен процес, като вместо десет групи, ще имаме три групи. По такъв начин ще получим следните групи от параметри: $POP^{FT}$, $POP^{HT}$ и $POP^{ST}$, за които ще разгледаме ограничения, наложени от изискването за решимост на математическата задача и за реализуемост на симулационното моделиране и оптимизационния процес.

Ограничения върху обекта за симулация. 1) Ограничения върху $POP^{FT}$. Ще припомним, че обобщеният съставен елемент FT' обединява съставните елементи FT, PO и SG и съдържа техните характерните компоненти. За него ще са в сила следните предположения: всички функции и задачи, изпълнявани от защитната система, трябва да могат да се представят чрез дискретно разпределен граф, чиито нива на паралелност са най-много 256 (F10); всяка функция и задача може да се раздели на отделни програмни елементи, така че броят на програмните елементи в едно ниво е най-много 32 (F06); всеки програмен елемент може да направи най-много 32 (F05) междинни програмни връзки, като съдържа задължително една начална и една крайна операция, а останалите операции могат да бъдат аритметични или логически, като техният брой и честота на поява ще се определят като входни данни на модела; 2) Ограничения върху $POP^{HT}$. Ще припомним, че обобщеният съставен елемент HT' обединява съставните елементи HT, DT и PH и съдържа техните характерните компоненти. Те са твърде много и е целесъобразно да бъдат обединени, за да може да се направи работоспособен симулационен модел. Обединението ще бъде под формата на следните ресурси: PRC, MEM и PRF. Тези ресурси са представени в избраните структури чрез съответни подсистеми, като за тях са в сила следните предположения: процесорната подсистема може да съдържа еднотипни (универсални) и многотипни (специализирани) PRC, като в рамките на един тип PRC са включени кристали с еднакви възможности (разрядност, скорост, инструкции и други); броят на типовете PRC за избраните структури ще ограничим съответно до 1 (универсален) за S11, до 2 (универсален и защитен) за S22 и до 4 (трансмисионно-контролен (tcp), мрежово-предаващ (ip), транспортно-информационен (udp) и контролно-предаващ (icmp)) за S33, като общото количество процесори във всяка от избраните структури е най-много 12; в случаите, когато работната памет и данновата памет са разделени на области, като всяка област има директна връзка с отделен процесор, броят на областите на които са разделени паметите е равен на броя на процесорите в структурата; периферните устройства, влизащи в състава на структурата, са минимум 3, като те включват "екран", "клавиатура" и "линия"; ресурсът управление е съсредоточен физически в програмната памет в резултат на някаква зареждаща процедура, която се осъществява еднократно за дадена функция и задача от управляващите server/host/mainframe/domain системи или локалната система, като освен това се предполага, че обемът на всички памети е достатъчен за извършване на операциите във всички програмни елементи на дадена функция и задача; 3) Ограничения върху $POP^{ST}$. Ще припомним, че обобщеният съставен елемент ST' обединява съставните елементи NC, PC, ST и PS. Поради своята многобройност и вътрешна сложност е много трудно и в определена степен почти невъзможно те да бъдат зададени в коректен математически вид и съответно реализирани чрез симулационен модел. Затова ще ги обединим под формата на един-единствен ресурс, наречен управление *MNG*, за който са в сила следните предположения: в рамките на

една функция и задача управлението ще се извършва като един прекъснат във времето процес, при което отделните времеви интервали са с дължина, определена от времето за изпълнение на отделен програмен елемент; освен това, редът, по който ще се изпълняват програмните елементи, ще се задава по време на развитието на процеса, при което ще предполагаме, че управлението е съсредоточено на едно място в програмната памет и времето за управление ще бъде пренебрежимо малко; образуваните опашки пред отделните ресурси ще бъдат обслужвани в съответствие със зададен приоритет, а ако няма такъв - по реда на пристигане; изброените по-горе съставящи на ресурса управление, с изключение на първите четири, ще бъдат считани за зададени и неизменни по време на изследванията, като с изменение на четирите начални съставящи ще можем да изследваме избраните структури при различни видове управления.

Следствия от ограниченията. В резултат на направените разсъждения и уговорки можем да запишем следните зависимости, касаещи параметрите: ( FT = променлива, HT = променлива, NC = променлива, DT = константа, ST = константа, PH = константа, PS = константа) (3.53) или ( FT=F(FT), HT=F(STR,CMP), NC=F(MNG)) (3.54) което, приложено към една целева функция на оптимизационен процес има вида: TFOP=F(FT,STR,CMP,MNG) (3.55). Изводът, който може да се направи от (3.55) е, че по такъв начин голямото многообразие от параметри, характеризиращи свойствата на CPS, се редуцира до четири параметъра, които ще се използват в симулационните изследвания, съгласно поставените цели и задачи.

*Съставяне на моделиращ алгоритъм (SM4-23).* Въз основа на формалните модели, описани по-горе, отнасящи се до избрания тип функции и задачи и избраните структури на защитната система, може да се направи основният симулиращ алгоритъм. Той има 6 основни части: PART1 – съдържа входните данни за съответния модел на избраната структура, отнасящи се до FT, STR, CMP, и MNG; PART2 – генерираща в съответствие със зададена функция на разпределение на вероятностите конкретния вариант на задачата за текущото проиграване на модела; PART3 – представяща работата на управлението по съставянето на дискретно разпределен граф с максимално ниво на паралелност и минимално време за изпълнение за конкретния вариант на задачата; PART4 – представяща работата на управлението по разпределяне на програмния елемент от дискретно разпределения граф по съответните ресурси; PART5 – представяща работата на SXX по изпълнение на текущия програмен елемент; PART6 – съдържаща подготовката на модела за следващо проиграване. Освен това, в моделиращия алгоритъм са включени: OMNG – организационно управление в PART3; CMNG – изчислително управление в PART4; CnE – брояч на експериментите в PART6; SUB5.3, SUB5.4 и SUB5.5 - представящи конкретното изпълнение на текущия програмен елемент за трите избрани структури; $MPRC^{xxx.k}_{tot\text{-}load}$ – модел на процесор от xxx-тия тип, отчитащ пълната му заетост (total load); $MPRC^{xxx.k}_{branch\text{-}load}$ – аналогичен модел, отчитащ заетостта на процесора за междинни програмни връзки; $MPRC^{xxx.k}_{cpo\text{-}load}$ – аналогичен модел, отчитащ заетостта на процесора за CPO; CnB – брояч на междинните програмни връзки; MAT1 – матрица с размери, еквивалентни на дискретно разпределения граф, съдържаща единици там, където има програмен елемент; $CRD^{mat}_{x}$ и $CRD^{mat}_{y}$ – координати по x и по y на матрицата; $SUB^{mod}_{1}$ – подмодел 1, отчитащ времето за чакане в QUE1; $SUB^{mod}_{2}$ – подмодел 2, отчитащ времето за чакане във вътрешната (с повишен приоритет) част на QUE1; MAT1.1 – матрица, в която се записват свободните и изпълнимите програмни елементи след анализа на MAT1 с цел по-лесно намиране на програмни елементи с максимален брой разклонения (междинни програмни връзки), като при наличие на условие за избор матрицата може да съществува в повече от един вариант, изменяйки се от индекс 1.1 до 1.N; CMNG1 – вид изчислително управление, при който се търси свободен и изпълним програмен елемент само в един ред на MAT1; CMNG2 – същото, но търсенето е в цялата MAT1 (не фигурира в явен запис); CMNG3 – трети вид, при който се оценява способността на програмния елемент да разклонява изчислителния процес; $MNG^{org}_{1}$ - управление на потребителската информация, известно под името управляващи байтове; $MNG^{org}_{2}$ – управляващи последователности; $MNG^{org}_{3}$ – управляващи макроси.

*Реализиране на работен модел (SM4-24).* Въз основа на разглежданията и допусканията направени по-горе, отнасящи се до формалните модели на обекта на симулацията, ще бъдат създадени и съответните работни (симулационни) модели. В съответствие с направения избор за език на симулационно моделиране, ще бъде използван GPSS/H. Имайки предвид структурата на основните

елементи на езика GPSS/H, може да се направи следното разпределение за използването им при описанието на формалните модели.

Симулационен модел на дискретно разпределен граф. Конкретното описание на дискретно разпределен граф и неговите варианти ще се извърши с помощта на MATRIX SAVE VALUE (MSV), представляващо матрично поле с размерност, отговаряща на тази на графа. Тъй като MSV ще бъде повече от едно, то ще имаме следното множество от матрични полета: $MSV=(MSV_i) \rightarrow i=1\div21$ (3.56) където MSV1 ще съдържа в себе си програмен елемент и елементите на матрицата ще са 0 или 1, ако няма или има съответно програмен елемент, и ще са 2 или 3, ако изпълнението на даден програмен елемент е приключило и съответно не е завършило; от MSV2 до MSV32 са матрици, съдържащи координатите на междинните програмни връзки за всеки програмен елемент, като техните елементи ще имат стойност 0, ако няма междинна програмна връзка, или $CRD_x^{mat}$ или $CRD_y^{mat}$ на този програмен елемент, от който идва програмната връзка; MSV33 ще съдържа номерата на процесорите, които са изпълнили програмния елемент, като елементите й ще са 0, ако на това място няма програмен елемент, или ще са със стойност $k$, равна на номера на процесор, изпълнил програмния елемент; MSV34 ще съдържа информация за разклоняващата способност на програмен елемент като елементите й ще имат стойност 0, ако на това място в дискретно разпределения граф няма програмен елемент, или ще имат стойност $r$, равна на броя на междинните програмни връзки, от които зависи даден програмен елемент; MSV35 – ще съдържа информация за времето на изпълнение на даден програмен елемент, като нейните елементи ще имат стойност 0, ако на това място в дискретно разпределения граф няма програмен елемент, или стойност $t^{pro}$, ако има програмен елемент; MSV36 ще съдържа свободни и изпълними програмни елементи, като ще служи за основа на $MNG_3^{comp}$. Съществуването на стохастични величини със съответните вероятностни разпределения от дискретен характер ще бъде отразено чрез FUNCTION (FUN), като основно ще имаме едно множество $FUN=(FUN_i) \rightarrow i=1\div6$ (3.57), където FUN1 ще задава времето за изпълнение на началната четяща операция; FUN2 ще задава времето за изпълнение на крайната записваща операция; FUN3 ще задава времето за изпълнение на една операция в програмен елемент; FUN4 ще задава броя на нивата, през които се осъществява междинната програмна връзка ($s$); FUN5 ще задава броя на междинните програмни връзки, влизащи / излизащи в един програмен елемент ($r$); FUN6 ще задава броя на програмните елементи в отделно ниво на паралелност ($m$). Освен това ще се използва VARIABLE (VAR), като нейното множество е следното: $VAR=(VAR_i) \rightarrow c=1\div6$ (3.58) където VAR1 ще задава броя на операциите в даден програмен елемент; VAR2 ще задава времето за осъществяване на IRO; VAR3 ще задава броя на началните и крайни операции, VAR4 ще задава броя на нивата за дискретно разпределения граф, VAR5 ще задава времето за осъществяване на PRO, VAR6 ще задава времето за осъществяване на IWO.

Симулационен модел на избрана структура. Конкретното описание на формалните модели на избраните структури ще се извърши с помощта на FACILITY (FAC). Тук също може да се групира множество, което има вида: $FAC=(FAC_i) \rightarrow i=1\div P$ (3.59) където $P$ ще задава броя на устройствата, които ще са различни за различните структури. Тъй като основната цел на изследването е оптимизиране на структурите по отношение на ресурсите PRC и MEM, както и на връзките между тях, ще съсредоточим вниманието си върху ресурса PRC, който е определящ в горните взаимоотношения. За тази цел, всеки PRC, влизащ в състава на структурата, ще бъде представен от няколко FAC (устройства), които ще отразяват поотделно участието на всеки PRC в различните операции. Тези устройства ще имат следния символичен запис: $MPRC_{tot}^{xxxk}$ – ще описва характеристиките на $k$-тия процесор от $xxx$-тия тип като цяло; $MPRC_{sro}^{xxxk}$ – същото, но за SRO; $MPRC_{iro}^{xxxk}$ – същото, но за IRO; $MPRC_{pro}^{xxxk}$ – същото, но за PRO; $MPRC_{iwo}^{xxxk}$ – същото, но за IWO; $MPRC_{fwo}^{xxxk}$ – същото, но за FWO. По аналогичен начин, на останалите ресурси от структурата като MEM, AMEM, $PRF^{bus}$, $PRF^{in}$ и $PRF^{out}$ ще съпоставим по едно устройство. С помощта на тези устройства става възможно натрупването на статистически данни, свързани с определяне заетостта на ресурсите, с броя на обслужените запитвания, с изчисляване средното време за обслужване на едно запитване и т.н. В тази връзка може да се каже, че те осигуряват статистическото представяне на ресурсите на структурите. Но тъй като в структурите има освен ресурси и опашки, то те също трябва да се опишат. Това ще стане с помощта на QUEUE (QUE), които също могат да бъдат

представени като множество, т.е. **QUE**=$\left(\mathbf{QUE}_i\right) \rightarrow i=1\div H$ (3.60) където H ще задава броя на опашките, които са различни за различните структури; **QUE1** ще представя QUE1; **QUE2** ще представя QUE2; **QUE3** ще представя QUE3; като наличието на вторични опашки на QUE2 и QUE3 ще се представя със съответните вторични опашки на **QUE2** и **QUE3**. Тъй като структурите ще бъдат изследвани в динамичен режим, ще използваме специален обект на GPSS/H, наречен TRANSACTION (TRA), чрез който ще можем да представим всеки PRC и ще се симулира неговата дейност по издаване на запитвания към различните FAC. Следователно ще имаме следното множество: TRA=$\left(\mathrm{TRA}_i\right) \rightarrow i=1\div q$ (3.61), където q задава броя на процесорите в структурата. Всеки TRA ще носи в себе си $\mathrm{CRD}_x^{mat}$ и $\mathrm{CRD}_y^{mat}$ на обработвания в момента програмен елемент и на програмните му връзки, времената за отделните операции, номера на процесора, чиято работа симулира в момента и други.

Симулационен модел на критерии за оценка. В избрания симулационен език съществуват обекти, които позволяват бърза и лесна оценка на продуктивността (PRD), производителността (PRF) и използваемостта (USB) на изследваната структура. Тези обекти се наричат STORAGE (STO) и могат да се обединят като множество: STO=$\left(\mathrm{STO}_i\right) \rightarrow i=1\div 5$ (3.62) където: STO1 ще служи за определяне на $\mathrm{PRD}^{rel}$ и $\mathrm{PRD}^{mid}$; STO2 – съответно на $\mathrm{PRF}_{tot}^{sys}$ и $\mathrm{USB}_{tot}^{sys}$; STO3 – съответно на $\mathrm{PRF}_{useful}^{sys}$ и $\mathrm{USB}_{useful}^{sys}$; STO4 – съответно на $\mathrm{PRF}_{r\text{-}useful}^{sys}$ и $\mathrm{USB}_{r\text{-}useful}^{sys}$; STO5 – съответно на $\mathrm{PRF}_{s\text{-}useful}^{sys}$ и $\mathrm{USB}_{s\text{-}useful}^{sys}$. Важен момент е обстоятелството, че транзакцията престоява толкова време в STO, колкото е продължителността на операцията, симулирана в момента, и освен това ще считаме, че при влизането си в STO, всяка транзакция носи толкова PRD, PRF и USB, колкото има симулираното в момента FAC. С помощта на симулационните обекти TABLE (TAB) ще бъде представена и една друга страна от динамиката на процесите, а именно, разпределението на времената за чакане, времената между пристигане на запитванията, скоростта на пристигане на нови запитвания и т.н.

*Проверка за достоверност (SM4-25).* Целта на тази проверка е да видим как работи създаденият симулационен модел и дали неговата работа отговаря на предположенията за работа на реалния обект, който в случая е една проектируема система. Това ще бъде направено, като проверим дали са в сила разглежданията, направени по-горе за всеки един от моделите на изследваните структури.

Модел на структура S11 (MODEL 1). Анализът на работата на модела ни показва, че за него са в сила следните взаимоотношения: $\mathrm{LOD}_k^{tot}=\mathrm{LOD}_k^{sro}+\mathrm{LOD}_k^{iro}+\mathrm{LOD}_k^{pro}+\mathrm{LOD}_k^{fwo}$ (3.63) $\mathrm{LOD}^{tot}=\sum_k^K \mathrm{LOD}_k^{tot}\mathrm{K}_k^{fpu}$ (3.64)

$\mathrm{SIR}_k^{tot}=\mathrm{SIR}_k^{sro}+\mathrm{SIR}_k^{iro}+\mathrm{SIR}_k^{pro}+\mathrm{SIR}_k^{fwo}$ (3.65) $\mathrm{SIR}^{tot}=\sum_{k=1}^K \mathrm{SIR}_k^{tot}$ (3.66) $\mathrm{SIR}_k^{sr\&fwo}=\mathrm{SIR}_k^{sro}+\mathrm{SIR}_k^{fwo}$ (3.67)

$\mathrm{SIR}_k^{iro:}=\mathrm{SIR}_{k,j}^{irq}+\mathrm{SIR}_{k,k}^{iro}$ (3.68) където нови означения са: $\mathrm{SIR}_k^{sr\&fwo}$ – обслужени запитвания от ресурса процесор от тип xxx с пореден номер k за начална четяща и FWO; $\mathrm{SIR}_{k,j}^{irq}$ – обслужени запитвания от ресурса процесор от тип xxx с пореден номер k при прекъсване на работата му от процесор от тип xxx с пореден номер j; $\mathrm{SIR}_{k,k}^{iro:}$ – обслужени запитвания от ресурса процесор от тип xxx с пореден номер k при IRO към същия k-ти процесор. При съпоставянето на горните изрази с развитието на процесите в S11, може да се направи изводът, че няма противоречие и резултатите от евентуални проиграванията на MODEL 1 могат да бъдат считани за достоверни.

Модел на структура S22 (MODEL 2). Проведеният по аналогичен начин анализ на работата на модела показва: $\mathrm{LOD}_k^{tot}=\mathrm{LOD}_k^{sro}+\mathrm{LOD}_k^{iro}+\mathrm{LOD}_k^{pro}+\mathrm{LOD}_k^{iwo}+\mathrm{LOD}_k^{fwo}$ (3.69) $\mathrm{LOD}^{tot}=\sum_k^K \mathrm{LOD}_k^{tot}\mathrm{K}^{fpu}$ (3.70)

$\mathrm{NSI}_k^{tot}=\mathrm{NSI}_k^{sro}+\mathrm{NSI}_k^{iro}+\mathrm{NSI}_k^{pro}+\mathrm{NSI}_k^{iwo}+\mathrm{NSI}_k^{fwo}$ (3.71) $\mathrm{NSI}^{tot}=\sum_{k=1}^K \mathrm{NSI}_k^{tot}$ (3.72) $\mathrm{NSI}_k^{sr\&fwo}=\mathrm{NSI}_k^{sro}+\mathrm{NSI}_k^{fwo}$ (3.73)

$\mathrm{NSI}_k^{pro}=\mathrm{NSI}_k^{iwo}$ (3.74) $\mathrm{NSI}_k^{iro}=\mathrm{NSI}_{k,k}^{iro}$ (3.75). При съпоставянето на горните изрази с развитието на процесите в S22, може да се направи изводът, че няма противоречие и резултатите от евентуални проиграванията на MODEL 2 могат да бъдат считани за достоверни.

Модел на структура S33 (MODEL 3). Аналогично и за този модел могат да се запишат следните съотношения: $LOD_k^{tot}=LOD_k^{src}+LOD_k^{src}+LOD_k^{prc}+LOD_k^{fix}$ (3.76) $LOD^{tot}=\sum_k^K LOD_k^{tot}K^{tpu}$ (3.77) $LOD_{amem\text{-}w.k}=LOD_{amem\text{-}w.k}^{write}+LOD_{amem\text{-}w.k}^{read}$ (3.78) $LOD_{amem\text{-}w}=\sum_{k\ 1}^K LOD_{amem\text{-}w.k}$ (3.79) $NSI_k^{tot}=NSI_k^{src}+NSI_k^{src}+NSI_k^{prc}+NSI_k^{fix}$ (3.80) $NSI^{tot}=\sum_{k\ 1}^K NSI_k^{tot}$ (3.81) $NSI_{amem\text{-}w.k}=NSI_{amem\text{-}w.k}^{write}+NSI_{amem\text{-}w.k}^{read}$ (3.82) $NSI_{amem}=\sum_{k\ 1}^K NSI_{amem\text{-}w.k}$ (3.83), където нови означения са: $LOD_{amem\text{-}w.k}$ – заетост на k-тата област от работната памет; $LOD_{amem\text{-}w.k}^{write}$ и $LOD_{amem\text{-}w.k}^{read}$ – аналогична заетост при запис и при четене; $NSI_{amem\text{-}w.k}$ – обслужвани запитвания на ресурса област от работната памет с пореден номер k; $NSI_{amem\text{-}w.k}^{write}$ и $NSI_{amem\text{-}w.k}^{read}$ – аналогично при запис и четене; $LOD_{amem\text{-}w}$ и $NSI_{amem\text{-}w}$ – заетост и обслужени запитвания за цялата работната памет (всички области). При съпоставянето на горните изрази с развитието на процесите в S33, може да се направи изводът, че няма противоречие и резултатите от евентуални проигравания на MODEL 3 могат да бъдат считани за достоверни.

## ГЛАВА ЧЕТВЪРТА

ПЛАНИРАНЕ НА ЕКСПЕРИМЕНТИТЕ (SM5-26). Изходна предпоставка за планирането ще бъде обстоятелството, че характеристиките на изследваните структури, в съответствие с направените разглеждания и ограничения, ще зависят само от четири фактора, т.е. целевата функция на оптимизационния процес ще има само четири аргумента. Основната цел при планирането на експериментите е да се намали до възможния минимум техният брой, като се осигури изпълнение на поставените задачи за изследване. Поради това, ще разгледаме поведението на всеки аргумент.

ЗАВИСИМОСТ ОТ ВИДА НА ЗАДАЧИТЕ (FT). Различните видове функции и задачи се различават по времето за изпълнение и по броя на междинните програмни връзки, които съдържа всяка от тях. Времето за изпълнение зависи преди всичко от продължителността на програмните връзки (измервана с брой тактови импулси) и затова ще изследваме структурите при четири вида продължителности, които ще означим като: B1 = 3 тактови импулса, B2 = 6 тактови импулса (2B1), B3 = 18 тактови импулса (3B2) и B4 = 36 тактови импулса (2B3). По такъв начин, този аргумент ще има четири степени на изменение.

ЗАВИСИМОСТ ОТ ВИДА НА СТРУКТУРАТА (STR). За целите на нашите изследвания като най-подходящи са избрани структурите S11, S22 и S33, което означава, че този аргумент ще има три степени на изменение, като съставът на всяка структура ще се изменя отделно.

ЗАВИСИМОСТ ОТ ВИДА НА СЪСТАВА (CMP). Тъй като определящият ресурс за постигане на максимална използваемост и производителност в избраните структури е процесорът, то под изменение на състава ще разбираме изменение броя на процесорите в структурата, като за останалите ресурси ще предполагаме, че са в достатъчно количество и че са неизменни. Броят на процесорите ще се изменя от 1 до 8 за S11, от 2 до 8 за S22 и от 4 до 8 за S33. Това означава, че този аргумент ще има осем, седем и пет степени на изменение за всяка структура съответно.

ЗАВИСИМОСТ ОТ ВИДА НА УПРАВЛЕНИЕТО (MNG). Ще използваме три вида изчислително управление: MNGC1, MNGC2 и MNGC3 и три вида организационно управление: MNGO1, MNGO2 и MNGO3, т.е. този аргумент ще има шест степени на изменение.

Направеното дотук общо планиране на експериментите показва, че максималният брой експерименти, описващ изменението на всички аргументи във всичките им степени, е 240. Ако приемем, че получаването на статистически устойчиви резултати изисква всеки експеримент да се проиграе няколко пъти, например 8, то тогава общата цифра на проиграванията ще бъде 1920 пъти и ако за всяко проиграване е необходимо около 30 минути, то ще са ни необходими около 960 часа компютърно време. Явно това е неприемливо и е необходимо да минимизираме техния брой, като извършим конкретно планиране. Имайки предвид, че най-често срещаните продължителности на междинните програмни връзки са B2 и B3, от съображения за намаляване на времето за проиграване, избираме B2. След определяне на оптималното q (общо (8+7+5)x3=60 експеримента), ще определим и оптималното B (общо (4+4+4)x3=36 експеримента), като така експериментите се намаляват общо до 96

(30`x96=2880`=48h), без това да пречи на изследванията. По аналогичен начин се провежда планирането и за организационното управление.

ВХОДНИ ДАННИ ЗА ЕКСПЕРИМЕНТИТЕ (SM5-27). Ще използваме вероятностното (0,2/X;0,3/Y;0,5/Z) задаване на случайни величини, които са използвани при формалните и програмни модели за определяне параметрите на дискретно разпределения граф на избрания тип функции и задачи, и параметрите на ресурсите на избраните за изследване структури на CPS.

(1) Време за изпълнение на начална четяща операция

$F_{N1}^{01}=FUN1.1=F_{N1}^{tsro}=2/6/4$ $F_{N2}^{01}=FUN1.2=F_{N2}^{tsro}=4/12/8$ $F_{N3}^{01}=FUN1.3=F_{N3}^{tsro}=12/36/24$ (4.1 a,b,c)

$F_{N4}^{01}=FUN1.4=F_{N4}^{tsro}=24/72/48$ $F_{N5}^{01}=FUN1.5=F_{N5}^{tsro}=72/216/144$ $F_{N6}^{01}=FUN1.6=F_{N6}^{tsro}=144/432/288$ (4.1 d,e,f)

$F_{N7}^{01}=FUN1.7=F_{N7}^{tsro}=432/1296/864$ $F_{N8}^{01}=FUN1.8=F_{N8}^{tsro}=864/2592/1728$ (4.1 g,h)

(2) Време за изпълнение на крайна записваща операция

$F_{N1}^{02}=FUN2.1=F_{N1}^{tfwo}=6/10/8$ $F_{N2}^{02}=FUN2.2=F_{N2}^{tfwo}=12/20/16$ $F_{N3}^{02}=FUN2.3=F_{N3}^{tfwo}=36/60/48$ (4.2 a,b,c)

$F_{N4}^{02}=FUN2.4=F_{N4}^{tfwo}=72/120/96$ $F_{N5}^{02}=FUN2.5=F_{N5}^{tfwo}=216/360/288$ $F_{N6}^{02}=FUN2.6=F_{N6}^{tfwo}=432/720/576$ (4.2 d,e,f)

$F_{N7}^{02}=FUN2.7=F_{N7}^{tfwo}=1296/2160/1728$ $F_{N8}^{02}=FUN2.8=F_{N8}^{tfwo}=2592/4320/3456$ (4.2 g,h)

(3) Време за изпълнение на една операция в един програмен елемент

$F_{N1}^{03}=FUN3.1=F_{N1}^{toc}=25/63/46$ $F_{N2}^{03}=FUN3.2=F_{N2}^{toc}=50/126/92$ $F_{N3}^{03}=FUN3.3=F_{N3}^{toc}=150/378/276$ (4.3 a,b,c)

$F_{N4}^{03}=FUN3.4=F_{N4}^{toc}=300/756/552$ $F_{N5}^{03}=FUN3.5=F_{N5}^{toc}=900/2268/1656$ $F_{N6}^{03}=FUN3.6=F_{N6}^{toc}=1800/4536/3312$ (4.3 d,e,f)

$F_{N7}^{03}=FUN3.7=F_{N7}^{toc}=5400/13608/9936$ $F_{N8}^{03}=FUN3.8=F_{N8}^{toc}=10800/27216/19872$ (4.3 g,h)

(4) Брой на нивата, през които се осъществява междинна програмна връзка

$F_{N1}^{04}=FUN4.1=F_{N1}^{nlb}=1/3/2$ $F_{N2}^{04}=FUN4.2=F_{N2}^{nlb}=2/4/3$ $F_{N3}^{04}=FUN4.3=F_{N3}^{nlb}=4/6/5$ (4.4 a,b,c)

$F_{N4}^{04}=FUN4.4=F_{N4}^{nlb}=5/7/6$ $F_{N5}^{04}=FUN4.5=F_{N5}^{nlb}=7/9/8$ $F_{N6}^{04}=FUN4.6=F_{N6}^{nlb}=8/10/9$ (4.4 d,e,f)

$F_{N7}^{04}=FUN4.7=F_{N7}^{nlb}=10/12/11$ $F_{N8}^{04}=FUN4.8=F_{N8}^{nlb}=11/13/12$ (4.4 g,h)

(5) Брой на програмните връзки, влизащи в един програмен елемент

$F_{N1}^{05}=FUN5.1=F_{N1}^{nbc}=3/7/5$ $F_{N2}^{05}=FUN5.2=F_{N2}^{nbc}=4/8/6$ $F_{N3}^{05}=FUN5.3=F_{N3}^{nbc}=6/10/8$ (4.5 a,b,c)

$F_{N4}^{05}=FUN5.4=F_{N4}^{nbc}=7/11/9$ $F_{N5}^{05}=FUN5.5=F_{N5}^{nbc}=9/13/11$ $F_{N6}^{05}=FUN5.6=F_{N6}^{nbc}=10/14/12$ (4.5 d,e,f)

$F_{N7}^{05}=FUN5.7=F_{N7}^{nbc}=12/16/14$ $F_{N8}^{05}=FUN5.8=F_{N8}^{nbc}=13/17/15$ (4.5 g,h)

(6) Брой на програмните елементи в отделен дискрет време

$F_{N1}^{06}=FUN6.1=F_{N1}^{ncd}=1/5/3$ $F_{N2}^{06}=FUN6.2=F_{N2}^{ncd}=2/6/4$ $F_{N3}^{06}=FUN6.3=F_{N3}^{ncd}=4/8/6$ (4.6 a,b,c)

$F_{N4}^{06}=FUN6.4=F_{N4}^{ncd}=5/9/7$ $F_{N5}^{06}=FUN6.5=F_{N5}^{ncd}=7/11/9$ $F_{N6}^{06}=FUN6.6=F_{N6}^{ncd}=8/12/10$ (4.6 d,e,f)

$F_{N7}^{06}=FUN6.7=F_{N7}^{ncd}=10/14/12$ $F_{N8}^{06}=FUN6.8=F_{N8}^{ncd}=11/15/13$ (4.6 g,h)

(7) Брой операции в програмен елемент

$F_{N1}^{07}=VAR1.1=F_{N1}^{noc}=2/6/4$ $F_{N2}^{07}=VAR1.2=F_{N2}^{noc}=3/7/5$ $F_{N3}^{07}=VAR1.3=F_{N3}^{noc}=5/9/7$ (4.7 a,b,c)

$F_{N4}^{07}=VAR1.4=F_{N4}^{noc}=6/10/8$ $F_{N5}^{07}=VAR1.5=F_{N5}^{noc}=8/12/10$ $F_{N6}^{07}=VAR1.6=F_{N6}^{noc}=9/13/11$ (4.7 d,e,f)

$F_{N7}^{07}=VAR1.7=F_{N7}^{noc}=4/5/4$ $F_{N8}^{07}=VAR1.8=F_{N8}^{noc}=12/16/14$ (4.7 g,h)

(8) Време за изпълнение на междинна четяща операция

$F_{N1}^{08}=VAR2.1=F_{N1}^{tiro}=1/3/2$ $F_{N2}^{08}=VAR2.2=F_{N2}^{tiro}=2/6/4$ $F_{N3}^{08}=VAR2.3=F_{N3}^{tiro}=6/18/12$ (4.8 a,b,c)

$F_{N4}^{08}=VAR2.4=F_{N4}^{tiro}=12/36/24$ $F_{N5}^{08}=VAR2.5=F_{N5}^{tiro}=36/108/72$ $F_{N6}^{08}=VAR2.6=F_{N6}^{tiro}=72/216/144$ (4.8 d,e,f)

$F_{N7}^{08}=VAR2.7=F_{N7}^{tiro}=216/648/432$ $F_{N8}^{08}=VAR2.8=F_{N8}^{tiro}=432/1296/864$ (4.8 g,h)

(9) Брой на началните и крайните операции

$F_{N1}^{09}=VAR3.1=F_{N1}^{nsfo}=0,40/1;0,60/2$ $F_{N2}^{09}=VAR3.2=F_{N2}^{nsfo}=0,35/1;0,65/2$ $F_{N3}^{09}=VAR3.3=F_{N3}^{nsfo}=0,30/1;0,70/2$ (4.9 a,b,c)

$F_{N4}^{09}=VAR3.4=F_{N4}^{nsfo}=0,25/1;0,75/2$ $F_{N5}^{09}=VAR3.5=F_{N5}^{nsfo}=0,20/1;0,80/2$ $F_{N6}^{09}=VAR3.6=F_{N6}^{nsfo}=0,15/1;0,85/2$ (4.9 d,e,f)

$F_{N7}^{09}=VAR3.7=F_{N7}^{nsfo}=0,10/1;0,90/2$ $F_{N8}^{09}=VAR3.8=F_{N8}^{nsfo}=0,05/1;0,95/2$ (4.9 g,h)

(10) Брой на нивата за дискретно разпределен граф

$F_{N1}^{10}=VAR4.1=F_{N1}^{nlg}=10/60/40$ $F_{N2}^{10}=VAR4.2=F_{N2}^{nlg}=20/70/50$ $F_{N3}^{10}=VAR4.3=F_{N3}^{nlg}=40/90/70$ (4.10 a,b,c)

$F_{N4}^{10}$=VAR4.4=$F_{N4}^{nlg}$=50/100/80 $F_{N5}^{10}$=VAR4.5=$F_{N5}^{nlg}$=70/120/100 $F_{N6}^{10}$=VAR4.6=$F_{N6}^{nlg}$=80/130/110 (4.10 d,e,f)

$F_{N7}^{10}$=VAR4.7=$F_{N7}^{nlg}$=100/150/130 $F_{N8}^{10}$=VAR4.8=$F_{N8}^{nlg}$=110/160/140 (4.10 g,h)

(11) Време за изпълнение на обработваща операция

$F_{N1}^{11}$=VAR5.1=$F_{N1}^{tpro}$=5/15/10 $F_{N2}^{11}$=VAR5.2=$F_{N2}^{tpro}$=10/30/20 $F_{N3}^{11}$=VAR5.3=$F_{N3}^{tpro}$=30/90/60 (4.11 a,b,c)

$F_{N4}^{11}$=VAR5.4=$F_{N4}^{tpro}$=60/180/120 $F_{N5}^{11}$=VAR5.5=$F_{N5}^{tpro}$=180/540/360 $F_{N6}^{11}$=VAR5.6=$F_{N6}^{tpro}$=360/1080/720 (4.11 d,e,f)

$F_{N7}^{11}$=VAR5.7=$F_{N7}^{tpro}$=1080/3240/2160 $F_{N8}^{11}$=VAR5.8=$F_{N8}^{tpro}$=2160/6480/4320 (4.11 g,h)

(12) Време за изпълнение на междинна записваща операция

$F_{N1}^{12}$=VAR6.1=$F_{N1}^{tiwo}$=4/8/6 $F_{N2}^{12}$=VAR6.2=$F_{N2}^{tiwo}$=8/16/12 $F_{N3}^{12}$=VAR6.3=$F_{N3}^{tiwo}$=24/48/36 (4.12 a,b,c)

$F_{N4}^{12}$=VAR6.4=$F_{N4}^{tiwo}$=48/96/72 $F_{N5}^{12}$=VAR6.5=$F_{N5}^{tiwo}$=144/288/216 $F_{N6}^{12}$=VAR6.6=$F_{N6}^{tiwo}$=288/576/432 (4.12 d,e,f)

$F_{N7}^{12}$=VAR6.7=$F_{N7}^{tiwo}$=864/1728/1296 $F_{N8}^{12}$=VAR6.8=$F_{N8}^{tiwo}$=1728/3456/2592 (4.12 g,h)

В горните изрази за по-голяма яснота са използвани означенията на функциите в програмните модели и във формалните модели. По такъв начин става възможно да се опишат характеристиките на дискретно разпределения граф, представящ отделни варианти на функции и задачи, като структура и като времеви съотношения. За по-пълното определяне на дискретно разпределения граф трябва да се използва и характеристиката периодичност на повторение на отделните видове изчислителни или логически операции, изпълнявани от процесорите на структурите. Това са: събиране, изваждане, умножение, деление, логически операции и условен преход. Тяхната периодичност на повторение за определен период от време се определя от т.нар. отношения на Морис [276, 282] или смеси на Гибсон [285, 290]. За нашите цели те ще имат следния вид: събиране и изваждане – 4/50/150; умножение – 3/20/300; логически операции и условен преход – 2/20/50; деление – 1/10/500. Първото число означава периодичност на повторение, второто – съответен процент и третото – продължителност на операциите в брой тактове. Тъй като е необходимо резултатите от моделирането да не се обвързват с конкретна реална структура на CPS, то ще се използва принципът на относителността при задаване на изчислителни мощности за отделните ресурси на изследваните структури. По такъв начин се постига универсалност при изследването, тъй като за всеки случай на изследване може да се дефинират различни изчислителни мощности. Това ще стане като определим тегловните съотношения между ресурсите на структурата с помощта на коефициенти. Логично е най-висок тегловен коефициент да има ресурсът процесор, тъй като преобладаващата част от дейността на структурите са изчислителни операции. Следващият по важност ресурс е работната памет и следователно нейният тегловен коефициент ще бъде близък до този на процесора. При сравнението между входната периферия и изходната периферия, по-голям тегловен коефициент ще се даде на изходната, тъй като ще считаме, че изходните (крайни) операции ще преобладават над входните (начални). Конкретните стойности на тегловните коефициенти за отделните ресурси са следните: процесор от xxx-ти тип ($PRC^{xxx}$) – 4; памет работна ($MEM^{work}$) – 3; периферия изходна ($PRF^{output}$) – 2; периферия входна ($PRF^{input}$) – 1.

ПРОИГРАВАНЕ НА МОДЕЛИТЕ И СЪБИРАНЕ НА РЕЗУЛТАТИТЕ (SM5-28). Проиграването на моделите и събирането на резултатите се извърши в Национална лаборатория по компютърна вирусология при БАН, като бе използвана наличната техника от типа PC Base. Симулационната програмна система е GPSS/H под управлението на операционна система MS Windows 98. Събирането на резултатите е осъществено чрез графики в Приложение 1 и чрез таблици в Приложение 2 на дисертацията, като по такъв начин се постига необходимото удобство, улесняващо анализа им.

АНАЛИЗ НА РЕЗУЛТАТИТЕ СПРЯМО АРГУМЕНТ СЪСТАВ (SM-29A). В съответствие с конкретното планиране на експериментите условията, при които са осъществени, са обобщени в Таблица 4.01. Изходните данни от експериментите са събрани в таблици, представени в Приложение 2 на дисертацията.

Таблица 4.01

| STR | S11 | S22 | S33 |
|---|---|---|---|
| FT | B=B2 | B=B2 | B=B2 |
| CMP | q=1-8 | q=2-8 | q=4-12 |
| MNG | C, O = 1-3 | C, O = 1-3 | C, O = 1-3 |

Изменението на критериите за оценка е показано на съответни графики, представени в Приложение 1 на дисертацията. Тяхното намаляване, увеличаване и екстремуми във функция от броя на процесорите е обобщено в Таблица 4.03.

Таблица 4.03

| $EVC^{com}$ | S11 | | | | S22 | | | | S33 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | C1÷C3 | | O1÷O3 | | C1÷C3 | | O1÷O3 | | C1÷C3 | | O1÷O3 | |
| | min | max | min | max | min | max | min | max | min | max | min | max |
| $TIM^{sys}$ | q=3 | - | q=3 | - | q=6 | - | q=6 | - | q=8 | - | q=6 | - |
| $SPD^{sys}$ | - | q=3 | - | q=3 | - | q=6 | - | q=6 | - | q=8 | - | q=6 |
| $PRD^{sys}$ | - | q=3 | - | q=3 | - | q=6 | - | q=6 | - | q=8 | - | q=6 |
| $PRF^{sys}$ | - | q=3 | - | q=3 | - | q=6 | - | q=6 | - | q=8 | - | q=6 |
| $USB^{sys}$ | q=8 | q=1 | q=8 | q=1 | q=8 | q=2 | q=8 | q=2 | q=12 | q=4 | q=12 | q=4 |
| $EVC^{spe}$ | S11 | | | | | | | | | | | |
| | C1 | | C2 | | C3 | | O1 | | O2 | | O3 | |
| | Min | max | min | max | min | max | min | max | min | max | min | max |
| $PRF^{prc}$ | q=1 | q=3 | q=1 | q=3 | q=1 | q=3 | q=1 | q=3 | q=1 | q=3 | q=1 | q=3 |
| $PRF^{mem}$ | q=1 | q=7 | q=1 | q=6 | q=1 | q=3 | q=1 | q=3 | q=1 | q=8 | q=1 | q=3 |
| $PRF^{prf}$ | q=1 | q=5 | q=1 | q=8 | q=1 | q=3 | q=1 | q=3 | q=1 | q=7 | q=1 | q=3 |
| $LOD^{prc}$ | q=8 | q=3 | q=8 | q=2 | q=1 | q=3 | q=8 | q=2 | q=8 | q=1 | q=8 | q=2 |
| $LOD^{mem}$ | q=2 | q=8 | q=2 | q=8 | q=3 | q=1 | q=2 | q=8 | q=1 | q=8 | q=2 | q=8 |
| $LOD^{prf}$ | q=2 | q=8 | q=3 | q=8 | q=4 | q=8 | q=2 | q=3 | q=1 | q=7 | q=3 | q=5 |
| $EVC^{spe}$ | S22 | | | | | | | | | | | |
| | C1 | | C2 | | C3 | | O1 | | O2 | | O3 | |
| | Min | max | min | max | min | max | min | max | min | max | min | max |
| $PRF^{prc}$ | q=2 | q=6 | q=2 | q=6 | q=2 | q=6 | q=2 | q=6 | q=2 | q=6 | q=2 | q=6 |
| $PRF^{mem}$ | q=2 | q=8 | q=2 | q=5 | q=2 | q=6 | q=2 | q=8 | q=2 | q=8 | q=2 | q=8 |
| $PRF^{prf}$ | q=2 | q=7 | q=2 | q=7 | q=2 | q=6 | q=2 | q=8 | q=2 | q=6 | q=2 | q=7 |
| $LOD^{prc}$ | q=8 | q=6 | q=8 | q=6 | q=6 | q=5 | q=8 | q=4 | q=8 | q=4 | q=8 | q=6 |
| $LOD^{mem}$ | q=6 | q=8 | q=3 | q=8 | q=5 | q=6 | q=4 | q=8 | q=4 | q=8 | q=6 | q=8 |
| $LOD^{prf}$ | q=6 | q=7 | q=6 | q=7 | q=5 | q=6 | q=4 | q=8 | q=4 | q=8 | q=6 | q=7 |
| $EVC^{spe}$ | S33 | | | | | | | | | | | |
| | C1 | | C2 | | C3 | | O1 | | O2 | | O3 | |
| | min | max | min | max | min | Max | min | max | min | max | min | max |
| $PRF^{prc}$ | q=4 | q=8 | q=4 | q=8 | q=4 | q=8 | q=4 | q=8 | q=4 | q=8 | q=4 | q=8 |
| $PRF^{mem}$ | q=4 | q=11 | q=4 | q=10 | q=4 | q=8 | q=4 | q=11 | q=4 | q=11 | q=4 | q=8 |
| $PRF^{prf}$ | q=4 | q=12 | q=4 | q=10 | q=4 | q=8 | q=4 | q=12 | q=4 | q=12 | q=4 | q=8 |
| $LOD^{prc}$ | q=12 | q=6 | q=6 | q=5 | q=5 | q=6 | q=12 | q=7 | q=12 | q=7 | q=8 | q=4 |
| $LOD^{mem}$ | q=6 | q=11 | q=4 | q=6 | q=4 | q=5 | q=7 | q=12 | q=7 | q=11 | q=4 | q=8 |
| $LOD^{prf}$ | q=6 | q=12 | q=5 | q=6 | q=6 | q=4 | q=7 | q=11 | q=7 | q=12 | q=4 | q=12 |
| EVC | S11 | | | | S22 | | | | S33 | | | |
| | C1÷C3 | | O1÷O3 | | C1÷C3 | | O1÷O3 | | C1÷C3 | | O1÷O3 | |
| | min | max | min | max | min | max | min | max | min | max | min | max |
| $NIS^{prc.mem.prf}$ | q=8 | q=1 | q=8 | q=1 | q=8 | q=2 | q=8 | q=2 | q=12 | q=4 | q=12 | q=4 |
| $TSI^{prc.mem.prf}$ | q=1 | q=8 | q=1 | q=8 | q=2 | q=8 | q=2 | q=8 | q=4 | q=12 | q=4 | q=12 |
| $NIQ^{que1.que2.que3}$ | q=1 | q=8 | q=1 | q=8 | q=2 | q=8 | q=2 | q=8 | q=4 | q=12 | q=4 | q=12 |
| $PZI^{que1.que2.que3}$ | q=8 | q=1 | q=8 | q=1 | q=8 | q=2 | q=8 | q=2 | q=12 | q=4 | q=12 | q=4 |
| $WTQ^{que1.que2.que3}$ | q=1 | q=8 | q=1 | q=8 | q=2 | q=8 | q=2 | q=8 | q=4 | q=12 | q=4 | q=12 |

Изводите, които могат да се направят, са следните: 1) Има увеличаване на системната скорост на обработка ($SPD^{sys}$), системната относителна продуктивност ($PRD^{sys}$), системната обща производителност ($PRF^{sys}$), производителността на подсистемите ($PRF^{prc.mem.prf}$) и заетостта на подсистемите ($LOD^{prc.mem.prf}$) при увеличаване броя на процесорите. Критериите достигат своя максимум при различен брой процесори за всяка една от структурите; 2) Има увеличаване на обслужения брой запитвания от подсистемите ($NSI^{prc.mem.prf}$), дължината на опашките пред подсистемите ($NIQ^{prc.mem.prf}$) и средното време за чакане на запитване в опашка ($WTQ^{prc.mem.prf}$) при увеличаване на броя на процесорите. Критериите достигат своя максимум при максимален брой процесори за всяка една от структурите; 3) Има намаляване на системното време за обработка ($TIM^{sys}$) при увеличаване на броя на процесорите. Критерият достига своя минимум при различен брой процесори за всяка една от структурите; 4) Има намаляване на системната обща използваемост ($USB^{sys}$), средното време за обслужване на запитване от подсистемите ($TSI^{prc.mem.prf}$) и процента на запитванията с нулево време на чакане в опашките ($PZI^{que1.que2.que3}$). Критериите достигат своя минимум при максимален брой процесори за всяка една от структурите; 5) Трябва да се предположи, че има връзка между броя на процесорите за всяка една от структурите и оптималните, в определен смисъл, стойности на критериите. За структурата S11 това е q=3, като вероятно обяснението за това е средната стойност на F04 - коефициента на вертикално разклонение в една от двете възможни посоки (3,55) на дискретно разпределения граф. За структурата S22 това е q=6, като вероятно обяснението за това е средната стойност на F05 - коефициента на входна мощност в една от двете възможни посоки (5,05) на програмен елемент. За структурата S33 това е q=8, като вероятно обяснението за това е F06 - коефициента на запълненост (8,2) на дискретно разпределения граф.

Препоръките за изследваните структури и управления, са следните: 1) При една практическа реализация на CPS, когато се определя размера на буферната й памет, нейното секциониране и съответния избор на елементната база, може да се използват получените резултати за дължината на опашките пред подсистемите, процентът на запитванията с нулево време на чакане в опашките и средното време за чакане на запитване в опашка; 2) Трябва да се посочи, че има слаба чувствителност на стойностите на критериите около максималния брой процесори за всяка една от структурите. Това показва, че няма нужда от експерименти с по-голям брой процесори.

АНАЛИЗ НА РЕЗУЛТАТИТЕ СПРЯМО АРГУМЕНТ ЗАДАЧИ (SM5-29B). В съответствие с конкретното планиране на експериментите, условията при които са осъществени, са показани в Таблица 4.04. Изходните данни от експериментите са събрани в таблици, представени в Приложение 2 на дисертацията.

Таблица 4.04

| STR | S11 | S22 | S33 |
|---|---|---|---|
| FT | B=B1÷B4 | B=B1÷B4 | B=B1÷B4 |
| CMP | q=3 | q=6 | q=8 |
| MNG | C, O = 1÷3 | C, O = 1÷3 | C, O = 1÷3 |

Изменението на критериите за оценка е показано на съответни графики, представени в Приложение 1 на дисертацията, като в Таблица 4.06 са описани екстремумите за четирите вида задачи.

Таблица 4.06

| EVC | S11 | | | | S22 | | | | S33 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | C1-C3 | | O1-O3 | | C1-C3 | | O1-O3 | | C1-C3 | | O1-O3 | |
| | min | max | min | max | min | max | min | max | min | max | min | max |
| $TIM^{sys}$ | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 |
| $SPD^{sys}$ | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 |
| $PRD^{sys}$ | B4 | - | B4 | - | B4 | - | B4 | - | B4 | - | B4 | - |
| $PRF^{sys}$ | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 |
| $USB^{sys}$ | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 |
| EVC | S11 | | | | S22 | | | | S33 | | | |
| | C1 | | O1 | | C1 | | O1 | | C1 | | O1 | |
| | min | max | min | max | min | max | min | max | min | max | min | max |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $PRF^{prc}$ | B4 | B1 | B1 | B2 | B4 | B2 | B1 | B4 | B1 | B4 | B1 | B4 |
| $PRF^{mem}$ | B1 | B4 | B1 | B2 | B1 | B4 | B1 | B3 | B1 | B3 | B1 | B3 |
| $PRF^{prf}$ | B1 | B3 | B1 | B2 | B1 | B3 | B1 | B4 | B1 | B3 | B1 | B4 |
| | C2 | | O2 | | C2 | | O2 | | C2 | | O2 | |
| $PRF^{prc}$ | B4 | B1 | B2 | B4 | B4 | B2 | B2 | B3 | B1 | B4 | B1 | B3 |
| $PRF^{mem}$ | B1 | B3 | B1 | B3 | B1 | B3 | B1 | B3 | B1 | B3 | B1 | B3 |
| $PRF^{prf}$ | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B3 | B1 | B3 | B1 | B4 |
| | C3 | | O3 | | C3 | | O3 | | C3 | | O3 | |
| $PRF^{prc}$ | B4 | B1 | B4 | B2 | B4 | B2 | B1 | B3 | B1 | B4 | B1 | B4 |
| $PRF^{mem}$ | B2 | B1 | B1 | B2 | B1 | B2 | B1 | B3 | B1 | B4 | B3 | B2 |
| $PRF^{prf}$ | B2 | B1 | B1 | B4 | B1 | B2 | B1 | B4 | B1 | B4 | B1 | B4 |
| EVC | S11 | | | | S22 | | | | S33 | | | |
| | C1 | | O1 | | C1 | | O1 | | C1 | | O1 | |
| | min | max | min | max | min | max | min | max | min | max | min | max |
| $LOD^{prc}$ | B4 | B1 | B2 | B1 | B4 | B1 | B4 | B1 | B3 | B1 | B4 | B1 |
| $LOD^{mem}$ | B1 | B4 | B1 | B2 | B1 | B4 | B2 | B3 | B1 | B3 | B4 | B2 |
| $LOD^{prf}$ | B1 | B3 | B1 | B2 | B1 | B3 | B1 | B4 | B1 | B3 | B1 | B4 |
| | C2 | | O2 | | C2 | | O2 | | C2 | | O2 | |
| $LOD^{prc}$ | B4 | B1 | B4 | B1 | B4 | B1 | B3 | B1 | B3 | B1 | B4 | B1 |
| $LOD^{mem}$ | B1 | B3 | B1 | B3 | B1 | B3 | B1 | B2 | B1 | B3 | B4 | B3 |
| $LOD^{prf}$ | B1 | B4 | B1 | B2 | B1 | B4 | B1 | B4 | B1 | B3 | B1 | B4 |
| | C3 | | O3 | | C3 | | O3 | | C3 | | O3 | |
| $LOD^{prc}$ | B4 | B2 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 |
| $LOD^{mem}$ | B2 | B4 | B1 | B2 | B1 | B3 | B1 | B3 | B1 | B4 | B3 | B2 |
| $LOD^{prf}$ | B2 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 |
| EVC | S11 | | | | S22 | | | | S33 | | | |
| | C1 | | O1 | | C1 | | O1 | | C1 | | O1 | |
| | min | max | min | max | min | max | min | max | min | max | min | max |
| $NIS^{prc}$ | B4 | B1 | B2 | B1 | B4 | B1 | B1 | B2 | B4 | B1 | B1 | B2 |
| $NIS^{mem}$ | B1 | B4 | B1 | B4 | B2 | B4 | B4 | B2 | B2 | B3 | B4 | B2 |
| $NIS^{prf}$ | B1 | B4 | B1 | B4 | B2 | B4 | B4 | B3 | B1 | B4 | B1 | B3 |
| | C2 | | O2 | | C2 | | O2 | | C2 | | O2 | |
| $NIS^{prc}$ | B3 | B1 | B2 | B4 | B4 | B1 | B3 | B2 | B4 | B1 | B1 | B2 |
| $NIS^{mem}$ | B2 | B4 | B4 | B2 | B2 | B4 | B4 | B2 | B2 | B4 | B4 | B2 |
| $NIS^{prf}$ | B1 | B2 | B3 | B2 | B2 | B4 | B4 | B2 | B2 | B1 | B4 | B2 |
| | C3 | | O3 | | C3 | | O3 | | C3 | | O3 | |
| $NIS^{prc}$ | B4 | B2 | B3 | B1 | B4 | B1 | B4 | B2 | B4 | B2 | B4 | B2 |
| $NIS^{mem}$ | B3 | B1 | B1 | B3 | B2 | B4 | B4 | B2 | B2 | B4 | B4 | B2 |
| $NIS^{prf}$ | B3 | B2 | B1 | B2 | B2 | B1 | B4 | B2 | B2 | B4 | B4 | B2 |
| EVC | S11 | | | | S22 | | | | S33 | | | |
| | C1 | | O1 | | C1 | | O1 | | C1 | | O1 | |
| | min | max | min | max | min | max | min | max | min | max | min | max |
| $TSI^{prc}$ | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 |
| $TSI^{mem}$ | B1 | B4 | B1 | B2 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 |
| $TSI^{prf}$ | B1 | B4 | B2 | B4 | B1 | B4 | B2 | B4 | B1 | B4 | B2 | B4 |
| | C2 | | O2 | | C2 | | O2 | | C2 | | O2 | |
| $TSI^{prc}$ | B1 | B4 | B1 | B4 | B1 | B4 | B2 | B4 | B1 | B4 | B2 | B4 |
| $TSI^{mem}$ | B1 | B4 | B1 | B4 | B1 | B2 | B2 | B4 | B1 | B2 | B1 | B4 |
| $TSI^{prf}$ | B2 | B4 | B2 | B4 | B1 | B4 | B2 | B4 | B1 | B4 | B2 | B4 |
| | C3 | | O3 | | C3 | | O3 | | C3 | | O3 | |
| $TSI^{prc}$ | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B2 | B4 | B2 | B4 |
| $TSI^{mem}$ | B1 | B4 | B2 | B4 | B1 | B4 | B2 | B4 | B1 | B4 | B2 | B4 |
| $TSI^{prf}$ | B2 | B4 | B2 | B4 | B1 | B4 | B2 | B4 | B1 | B2 | B2 | B4 |

| EVC | S11 | | | | S22 | | | | S33 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | C1 | | O1 | | C1 | | O1 | | C1 | | O1 | |
| | min | max | min | max | min | max | min | max | min | max | min | max |
| $NIQ^{prc}$ | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 |
| $NIQ^{mem}$ | B1 | B4 | B1 | B4 | B4 | B1 | B1 | B4 | B4 | B1 | B1 | B4 |
| $NIQ^{prf}$ | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 |
| | C2 | | O2 | | C2 | | O2 | | C2 | | O2 | |
| $NIQ^{prc}$ | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 |
| $NIQ^{mem}$ | B1 | B4 | B1 | B4 | B4 | B1 | B1 | B4 | B4 | B1 | B1 | B4 |
| $NIQ^{prf}$ | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 |
| | C3 | | O3 | | C3 | | O3 | | C3 | | O3 | |
| $NIQ^{prc}$ | B1 | B4 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 |
| $NIQ^{mem}$ | B1 | B4 | B1 | B4 | B4 | B1 | B1 | B4 | B4 | B1 | B1 | B4 |
| $NIQ^{prf}$ | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 |
| EVC | S11 | | | | S22 | | | | S33 | | | |
| | C1 | | O1 | | C1 | | O1 | | C1 | | O1 | |
| | min | max | min | max | min | max | min | max | min | max | min | max |
| $PZI^{prc}$ | B1 | B4 | B1 | B4 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 |
| $PZI^{mem}$ | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 |
| $PZI^{prf}$ | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 |
| | C2 | | O2 | | C2 | | O2 | | C2 | | O2 | |
| $PZI^{prc}$ | B1 | B4 | B1 | B4 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 |
| $PZI^{mem}$ | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 |
| $PZI^{prf}$ | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B4 | B4 | B1 |
| | C3 | | O3 | | C3 | | O3 | | C3 | | O3 | |
| $PZI^{prc}$ | B1 | B4 | B1 | B4 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 |
| $PZI^{mem}$ | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 |
| $PZI^{prf}$ | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 |
| EVC | S11 | | | | S22 | | | | S33 | | | |
| | C1-C3 | | O1-O3 | | C1-C3 | | O1-O3 | | C1-C3 | | O1-O3 | |
| | min | max | min | max | min | max | min | max | min | max | min | max |
| WTQ | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 | B1 | B4 |

Изводите, които могат да се направят, са следните: 1) Има увеличаване на системното време за обработка ($TIM^{sys}$), системната обща производителност ($PRF^{sys}$), системната обща използваемост ($USB^{sys}$) и средното време за чакане на запитване в опашка ($WTQ_{mid}^{que1.que2.que3}$) при изменение на видовете задачи от B1 до B4 за всяка една от структурите. Критериите за оценка достигат своя максимум при достигане на максималната стойност (B4) за трите структури; 2) Има намаляване на системната скорост на обработка ($SPD^{sys}$) и системната относителна продуктивност ($PRD^{sys}$) при изменение на видовете задачи от B1 до B4 за всяка една от структурите. Критериите за оценка достигат своя минимум при достигане на максималната стойност (B4) за трите структури; 3) Има различни изменения на производителността на подсистемите ($PRF^{prc.mem.prf}$), заетостта на подсистемите ($LOD^{prc.mem.prf}$), обслужения брой запитвания от подсистемите ($NSI^{prc.mem.prf}$), средното време за обслужване на запитване от подсистемите ($TSI^{prc.mem.prf}$), дължината на опашките пред подсистемите ($NIQ^{que1.que2.que3}$) и процента на запитванията с нулево време на чакане в опашките ($PZI^{que1.que2.que3}$) при изменение на видовете задачи от B1 до B4 за всяка една от структурите. Критериите за оценка достигат своя екстремум при достигане на различни стойности на B за трите структури; 4) Първият вид задачи (FT=B1) осигурява оптимални в определен смисъл стойности на критериите за оценка в следните случаи: 1. За всички изследвани структури и управления по отношение на: системното време на обработка ($TIM^{sys}$), системната скорост на обработка ($SPD^{sys}$), системната относителна продуктивност ($PRD^{sys}$) и средното време за чакане на запитване в опашка ($WTQ^{que1.que2.que3}$); 2. За първата структура с третото изчислително управление, по отношение на производителността на подсистемите ($PRF^{prc.mem.prf}$); 3. За първата структура при всички изчислителни и при второ и трето организационни управления; за втората структура при първо и трето изчислително, и при всички организационни управления; за

третата структура при първо изчислително и при всички организационни управления по отношение на средното време на обслужване на запитване от подсистемите ($TSI^{prc.mem.prf}$); 4. За втората и третата структура при всички изчислителни и организационни управления по отношение на процента на запитванията с нулево време на чакане в опашките ($PZI^{que1.que2.que3}$). 5) Вторият вид задачи (FT=B2) осигурява оптимални, в определен смисъл, стойности на критериите за оценка в следните случаи: 1. За първата структура с първо организационно управление и за втората структура с трето изчислително управление по отношение на производителността на подсистемите ($PRF^{prc.mem.prf}$); 2. За втората и третата структура с всички организационни управления по отношение на обслужения брой запитвания от подсистемите ($NSI^{prc.mem.prf}$). 6) Третият вид задачи (FT=B3) осигурява оптимални в определен смисъл стойности на критериите за оценка за втората структура с второ организационно управление по отношение на производителността на подсистемите ($PRF^{prc.mem.prf}$); 7) Четвъртият вид задачи (FT=B4) осигурява оптимални в определен смисъл стойности на критериите за оценка в следните случаи: 1. За всички структури с всички управления, по отношение на системната обща производителност ($PRF^{sys}$) и системната обща използваемост ($USB^{sys}$); 2. За третата структура с третото изчислително и третото организационно управление по отношение на производителността на подсистемите ($PRF^{prc.mem.prf}$).

Препоръките, които могат да се направят за изследваните структури и управления, са следните: 1) При първата структура трябва да се има предвид нейната особеност, а именно увеличаване броя на прекъсванията при увеличаване броя на процесорите. Определянето на оптималния в определен смисъл тип задачи за първата структура трябва да става въз основа на броя на прекъсванията на един програмен елемент, времето за изпълнение на един програмен елемент, общия брой на междинните програмни връзки, които се съдържат в програмния елемент, и дискретно разпределения граф на задачата като цяло; 2) При втората структура трябва да се има предвид нейната конфликтна точка – опашка 3. Определянето на оптималния тип задачи за втората структура трябва да става въз основа на характеристиките на задачите, свързани с броя на междинните програмни връзки, времето за изпълнението им, процентното съдържание на времето за изпълнение на междинните програмни връзки в общото време за изпълнение на програмния елемент. Необходимо е да се отчете, че характеристиките на опашка 3 са отражение на броя на програмните елементи в един дискрет време на дискретно разпределения граф на задачата и на сумата от всички междинни програмни връзки, влизащи в един програмен елемент за един отделен дискрет време от изчислителния процес; 3) При третата структура трябва да се има предвид повишената специализация на процесорите, преките връзки процесор - памет и наличието на локална памет. Следователно, най-добрите задачи за тази структура са тези, които имат подходящо профилиране за всеки отделен тип процесор.

АНАЛИЗ НА РЕЗУЛТАТИТЕ СПРЯМО АРГУМЕНТ УПРАВЛЕНИЕ (SM5-29C). В съответствие с конкретното планиране на експериментите, условията, при които са осъществени, са показани на Таблица 4.07. Изходните данни от експериментите са събрани в таблици, представени в Приложение 2 на дисертацията, като тяхното групиране по структури и видове критерии за оценка е дадено в Таблица 4.02. Изменението на критериите за оценка е показано на съответни графики, представени в Приложение 1 на дисертацията, като в Таблица 4.08 са описани екстремумите за шестте вида управления.

Таблица 4.07

| STR | S11 | S22 | S33 |
|---|---|---|---|
| FT | B=B2 | B=B2 | B=B2 |
| CMP | q=3 | q=6 | q=8 |
| MNG | C, O = 1÷3 | C, O = 1÷3 | C, O = 1÷3 |

Таблица 4.08

| EVC | S11 | | | | S22 | | | | S33 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | C1÷C3 | | O1÷O3 | | C1÷C3 | | O1÷O3 | | C1÷C3 | | O1÷O3 | |
| | min | max | min | max | min | max | min | max | min | max | min | max |
| $TIM^{sys}$ | C=3 | C=1 | O=1 | O=3 | C=3 | C=1 | O=2 | O=1 | C=3 | C=1 | O=3 | O=1 |
| $SPD^{sys}$ | C=1 | C=3 | O=3 | O=1 | C=1 | C=3 | O=1 | O=2 | C=1 | C=3 | O=1 | O=3 |
| $PRD^{sys}$ | C=1 | C=3 | O=3 | O=1 | C=1 | C=3 | O=1 | O=2 | C=1 | C=3 | O=1 | O=3 |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $PRF^{sys}$ | C=1 | C=3 | O=3 | O=1 | C=1 | C=3 | O=1 | O=2 | C=1 | C=3 | O=1 | O=3 |
| $USB^{sys}$ | C=1 | C=3 | O=3 | O=1 | C=1 | C=3 | O=1 | O=2 | C=1 | C=3 | O=1 | O=3 |
| $PRF^{prc}$ | C=1 | C=3 | O=3 | O=1 | C=1 | C=3 | O=1 | O=2 | C=1 | C=3 | O=1 | O=3 |
| $PRF^{mem}$ | C=1 | C=3 | O=3 | O=1 | C=1 | C=3 | O=2 | O=3 | C=2 | C=3 | O=2 | O=3 |
| $PRF^{prf}$ | C=1 | C=3 | O=3 | O=1 | C=1 | C=3 | O=3 | O=2 | C=1 | C=3 | O=2 | O=3 |
| $LOD^{prc}$ | C=1 | C=3 | O=3 | O=1 | C=1 | C=2 | O=1 | O=2 | C=1 | C=3 | O=1 | O=3 |
| $LOD^{mem}$ | C=3 | C=1 | O=1 | O=3 | C=2 | C=1 | O=2 | O=1 | C=3 | C=1 | O=3 | O=2 |
| $LOD^{prf}$ | C=3 | C=1 | O=2 | O=1 | C=2 | C=3 | O=3 | O=2 | C=3 | C=1 | O=3 | O=2 |
| $NIS^{prc}$ | C=1 | C=3 | O=1 | O=3 | C=1 | C=3 | O=2 | O=1 | C=3 | C=2 | O=3 | O=1 |
| $NIS^{mem}$ | C=2 | C=1 | O=1 | O=3 | C=2 | C=3 | O=1 | O=2 | C=2 | C=3 | O=3 | O=1 |
| $NIS^{prf}$ | C=1 | C=3 | O=2 | O=3 | C=2 | C=1 | O=2 | O=3 | C=1 | C=2 | O=3 | O=1 |
| $TSI^{prc.mem.prf}$ | C=3 | C=1 | O=1 | O=3 | C=3 | C=1 | O=2 | O=1 | C=3 | C=1 | O=3 | O=1 |
| $NIQ^{que1.que2.que3}$ | C=3 | C=2 | O=1 | O=3 | C=3 | C=2 | O=2 | O=1 | C=3 | C=2 | O=3 | O=1 |
| $PZI^{que1.que2.que3}$ | C=1 | C=3 | O=3 | O=1 | C=1 | C=3 | O=3 | O=2 | C=1 | C=3 | O=1 | O=3 |
| $WTQ^{que1.que2.que3}$ | C=3 | C=1 | O=1 | O=3 | C=3 | C=1 | O=2 | O=1 | C=3 | C=1 | O=3 | O=1 |

Изводите, които могат да се направят, са следните: 1) Има увеличаване на системната скорост на обработка ($SPD^{sys}$), системната относителна продуктивност ($PRD^{sys}$), системната обща производителност ($PRF^{sys}$), системната обща използваемост ($USB^{sys}$) и процента на запитванията с нулево време на чакане в опашките ($PZI^{que1.que2.que3}$) при изменение на видовете изчислителни управления. Критериите за оценка достигат своя максимум при достигане на максималната стойност (C3) за трите структури; 2) Има намаляване на системното време за обработка ($TIM^{sys}$), средното време за обслужване на запитване от подсистемите ($TSI^{prc.mem.prf}$) и средното време за чакане на запитване в опашки ($WTQ^{que1.que2.que3}$) при изменение на видовете изчислителни управления. Критериите за оценка достигат своя минимум при достигане на максималната стойност (C3) за трите структури; 3) Има различни изменения на дължината на опашките пред подсистемите ($NIQ^{que1.que2.que3}$) при изменение на видовете изчислителни управления. Критерият за оценка достига своя максимум при средната стойност (C2) за трите структури; 4) Има различни изменения на производителността на подсистемите ($PRF^{prc.mem.prf}$), заетостта на подсистемите ($LOD^{prc.mem.prf}$) и обслужения брой запитвания от подсистемите ($NSI^{prc.mem.prf}$) при изменение на видовете изчислителни управления. Критериите за оценка достигат своя екстремум при различни стойности на C за трите структури; 5) Има изменения на системното време на обработка ($TIM^{sys}$), системната скорост на обработка ($SPD^{sys}$), системната относителна продуктивност ($PRD^{sys}$), системната обща производителност ($PRF^{sys}$), системната обща използваемост ($USB^{sys}$), производителността на подсистемите ($PRF^{prc.mem.prf}$), заетостта на подсистемите ($LOD^{prc.mem.prf}$), обслужения брой запитвания от подсистемите ($NSI^{prc.mem.prf}$), средното време за обслужване на запитване от подсистемите ($TSI^{prc.mem.prf}$), дължината на опашките пред подсистемите ($NIQ^{que1.que2.que3}$), процента на запитванията с нулево време на чакане в опашките ($PZI^{que1.que2.que3}$) и средното време за чакане на запитване в опашка ($WTQ^{que1.que2.que3}$) при изменение на видовете организационни управления. Критериите за оценка достигат своя екстремум при различни стойности на O за трите структури; 6) След анализ на получените стойности за системното време за обработка ($TIM^{sys}$), системната скорост за обработка ($SPD^{sys}$), системната относителна продуктивност ($PRD^{sys}$), системната обща производителност ($PRF^{sys}$), системната обща използваемост ($USB^{sys}$), производителността на подсистемите ($PRF^{prc.mem.prf}$), заетостта на подсистемите ($LOD^{prc.mem.prf}$), обслужения брой запитвания от подсистемите ($NSI^{prc.mem.prf}$), дължината на опашките пред подсистемите ($NIQ^{prc.mem.prf}$), процента на запитванията с нулево време на чакане в опашките ($PZI^{que1.que2.que3}$) и средното време за чакане на запитване в опашка ($WTQ^{que1.que2.que3}$) може да се направи оценка за използването на трите типа организационно управление, съответно: първи тип (O1) – при първата структура, втори тип (O2) – при втората структура и трети тип (O3) – при третата структура.

Препоръките, които могат да се направят, са следните: 1) Изчислителното управление влияе най-съществено върху критериите за оценка поради влиянието на междинните програмни връзки; 2) При използване на изчислително управление за първата структура е необходимо да се прояви прецизност, тъй като този тип управление оказва пряко влияние върху увеличаване броя на прекъсванията на процесорите; 3) При използване на трите типа изчислително управление налице е предимство на третия

тип (C3) спрямо останалите; 4) При използване на трите типа организационно управление налице е пригодност на всеки тип към съответна структура, а именно: първи тип (O1) за първата структура, втори тип (O2) за втора структура и трети тип (O3) за трета структура.

АНАЛИЗ НА РЕЗУЛТАТИТЕ СПРЯМО АРГУМЕНТ СТРУКТУРА (SM5-29D). В съответствие с конкретното планиране на експериментите, условията, при които са осъществени, са показани на Таблица 4.09. Изходните данни от експериментите са събрани в таблици, представени в Приложение 2 на дисертацията, като тяхното групиране е описано в Таблица 4.02. Изменението на критериите за оценка е показано на съответни графики, представени в Приложение 1 на дисертацията, като в Таблица 4.10 са описани екстремумите за трите вида структури.

Таблица 4.09

| MNG | C=1, O=1 | C=2, O=2 | C=3, O3 |
|---|---|---|---|
| FT | B=B2 | B=B2 | B=B2 |
| CMP | q=6.8 | q=6.8 | q=6,8 |
| STR | S11÷S33 | S11÷S33 | S11÷S33 |

Таблица 4.10

| EVC | MNG1 | | | | MNG2 | | | | MNG3 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | C1 | | O1 | | C2 | | O2 | | C3 | | O3 | |
| | min | max | min | max | min | max | min | max | min | max | min | max |
| $TIM^{sys}$ | S33 | S11 | S11 | S33 | S33 | S11 | S22 | S11 | S33 | S11 | S33 | S11 |
| $SPD^{sys}$ | S11 | S33 | S33 | S11 | S11 | S33 | S11 | S22 | S11 | S33 | S11 | S33 |
| $PRD^{sys}$ | S11 | S33 | S33 | S11 | S11 | S33 | S11 | S22 | S11 | S33 | S11 | S33 |
| $PRF^{sys}$ | S11 | S33 | S33 | S11 | S11 | S33 | S11 | S22 | S11 | S33 | S11 | S33 |
| $USB^{sys}$ | S11 | S33 | S33 | S11 | S11 | S33 | S11 | S22 | S11 | S33 | S11 | S33 |
| $PRF^{prc}$ | S11 | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S33 | S11 |
| $PRF^{mem}$ | S33 | S11 | S33 | S22 | S33 | S11 | S33 | S22 | S33 | S11 | S33 | S22 |
| $PRF^{prf}$ | S11 | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S11 | S33 |
| $LOD^{prc}$ | S11 | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S11 | S33 |
| $LOD^{mem}$ | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S11 |
| $LOD^{prf}$ | S33 | S11 | S22 | S33 | S22 | S33 | S33 | S11 | S33 | S11 | S33 | S11 |
| $NIS^{prc}$ | S11 | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S33 | S22 |
| $NIS^{mem}$ | S33 | S11 | S11 | S33 | S33 | S22 | S11 | S33 | S11 | S22 | S33 | S22 |
| $NIS^{prf}$ | S33 | S11 | S11 | S33 | S33 | S22 | S11 | S33 | S33 | S11 | S33 | S22 |
| $TSI^{prc}$ | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S11 |
| $TSI^{mem}$ | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S11 | S33 | S11 |
| $TSI^{prf}$ | S33 | S11 | S33 | S22 | S33 | S11 | S22 | S11 | S33 | S11 | S33 | S11 |
| $NIQ^{que1}$ | S33 | S11 | S11 | S33 | S33 | S11 | S22 | S11 | S33 | S11 | S33 | S11 |
| $NIQ^{que2}$ | S33 | S11 | S11 | S33 | S33 | S11 | S11 | S22 | S33 | S11 | S33 | S11 |
| $NIQ^{que3}$ | S33 | S22 | S11 | S22 | S33 | S22 | S22 | S11 | S33 | S22 | S33 | S11 |
| $PZI^{que1}$ | S11 | S33 | S33 | S11 | S11 | S33 | S11 | S22 | S11 | S33 | S11 | S33 |
| $PZI^{que2}$ | S11 | S33 | S33 | S11 | S11 | S33 | S11 | S22 | S11 | S33 | S11 | S33 |
| $PZI^{que3}$ | S22 | S33 | S33 | S11 | S22 | S33 | S11 | S22 | S22 | S33 | S11 | S33 |
| $WTQ^{que1}$ | S33 | S22 | S11 | S33 | S33 | S22 | S22 | S11 | S33 | S22 | S33 | S11 |
| $WTQ^{que2}$ | S33 | S11 | S11 | S33 | S33 | S11 | S22 | S11 | S33 | S11 | S33 | S11 |
| $WTQ^{que3}$ | S33 | S22 | S11 | S33 | S33 | S22 | S22 | S11 | S33 | S22 | S33 | S11 |

Изводите, които могат да се направят, са следните: 1) Има увеличаване на системната скорост на обработка ($SPD^{sys}$), системната относителна продуктивност ($PRD^{sys}$), системната обща производителност ($PRF^{sys}$) и системната обща използваемост ($USB^{sys}$) при изменение на изчислителните управления за изследваните структури. Критериите за оценка достигат своя максимум при третата структура и за трите вида управления; 2) Има намаляване на системното време за обработка ($TIM^{sys}$) и средното време за обслужване на запитване от подсистемите ($TSI^{prc.mem.prf}$) при изменение на

изчислителните управления за изследваните структури. Критериите за оценка достигат своя минимум при третата структура за трите вида управления; 3) Има изменения на производителността на подсистемите ($PRF^{prc.mem.prf}$), заетостта на подсистемите ($LOD^{prc.mem.prf}$), обслужения брой запитвания от подсистемите ($NSI^{prc.mem.prf}$), дължината на опашките пред подсистемите ($NIQ^{prc.mem.prf}$), процента на запитванията с нулево време на чакане в опашките ($PZI^{que1.que2.que3}$) и средното време за чакане на запитване в опашка ($WTQ^{que1.que2.que3}$) при изменение на изчислителните управления за изследваните структури. Критериите за оценка достигат своя екстремум при различните структури за трите вида управления; 4) Има изменения на системното време за обработка ($TIM^{sys}$), системната скорост на обработка ($SPD^{sys}$), системната относителна продуктивност ($PRD^{sys}$), системната обща производителност ($PRF^{sys}$), системната обща използваемост ($USB^{sys}$), производителността на подсистемите ($PRF^{prc.mem.prf}$), заетостта на подсистемите ($LOD^{prc.mem.prf}$), обслужения брой запитвания от подсистемите ($NSI^{prc.mem.prf}$), средното време за обслужване на запитване от подсистемите ($TSI^{prc.mem.prf}$), дължината на опашките пред подсистемите ($NIQ^{prc.mem.prf}$), средното време за чакане на запитване в опашка ($WTQ^{que1.que2.que3}$) и процента на запитванията с нулево време на чакане в опашките ($PZI^{que1.que2.que3}$) при изменение на организационните управления за изследваните структури. Критериите за оценка достигат своя екстремум при различни структури за трите вида управления; 5) Най-добри резултати показва третата структура по отношение на системното време за обработка ($TIM^{sys}$), системната скорост на обработка ($SPD^{sys}$), системната относителна продуктивност ($PRD^{sys}$), системната обща производителност ($PRF^{sys}$), системната обща използваемост ($USB^{sys}$) и другите критерии за оценка. Най-слаби резултати по отношение на същите критерии показва първата структура.

Препоръките, които могат да се направят, са : изборът на оптимална структура от гледна точка на потребителя или на проектанта може да се извърши чрез минимизиране на целева функция. Потребителят би могъл да прецени относителната печалба по време използвайки оптималният брой процесори за всяка от изследваните структури или относителната загуба от неполезна работа (междинни програмни връзки). По подобен начин, изхождайки от своите конкретни цели, потребителят може да избере и някои от другите критерии за оценка, които да свърже с цената на обработка и чрез конкретна целева функция да избере оптимално решение. Проектантът би могъл да подбере съответни критерии за оценка, като има предвид типа на задачите, за които се проектира CPS, с цел максимална използваемост на ресурсите й. Подходящи критерии са системната обща полезна и неполезна производителност, системната обща полезна и неполезна използваемост, производителността, заетостта, обслужеият брой запитвания и средното време за обслужване на запитване от подсистемите за началните и крайните операции (SFO=M01), обработващи операции (PRO=M03) и междинните операции (IMO=M02), както и дължината на опашките, средното време за чакане на запитване, процентът на запитванията с нулево време на чакане в опашките и други.

## ЗАКЛЮЧЕНИЕ

По-съществените резултати от настоящата дисертационна работа могат да се обобщят по следния начин:

(1) С помощта на съществуващата информация в достъпните литературни източници е предложена класификация на CPS, основаваща се на три основни елемента: даннова, компютърна и комуникационна защита, като заедно с това е направен обзор на известните методи, модели и проблеми, имащи отношение към задачата за оптимално проектиране на CPS, при което са посочени техните предимства и недостатъци;

(2) Въз основа на този анализ е формулирана постановката за решаване на задачата за оптимално проектиране на CPS чрез съвместно използване на аналитично и симулационно изследване, при което, при зададени тип на изпълняваните функции и задачи и тип на структурата на CPS се извършва оценка на нейните характеристики, като в съответствие с това се прилага оптимизационен процес;

(3) Определени са еталонни структури, характеризиращи се с различна организация на паметта, с различна организация на връзките процесор - памет, с различен брой еднотипни копроцесори и с различен брой типове функционално специализирани копроцесори, като с цел минимизиране на изчислителните разходи при симулационното изследване са избрани и съответно моделирани само три структури: структура с универсален процесор, структура със специализиран защитен копроцесор и структура със специализирани защитни копроцесори;

(4) Предложени са критерии за оценка на характеристиките на CPS, които са разделени на две групи: общи – в които влизат време за обработка, скорост за обработка, цена за обработка от определена система за дадена задача и специфични: средна и относителна продуктивност, реална, относителна и обща производителност, реална, относителна и обща заетост, като са определени съотношенията между тях в зависимост от параметрите на избраните структури;

(5) В рамките на симулационното изследване са разработени алгоритми за изследване на основните характеристики на CPS, като заедно с това са избрани източниците на случайни величини и вероятностните функции, използвани при симулацията;

(6) Въз основа на направените разглеждания е създаден и използван програмен пакет за симулационно моделиране на CPS, състоящ се от отделни симулационни програми за всяка от избраните структури;

(7) Програмният пакет за симулационно моделиране на CPS е използван при осъществяването на следните симулационни експерименти:

1) Изследване влиянието на структурата, състава и управлението на CPS върху нейните характеристики при даден тип задачи.

2) Изследване влиянието на типа задачи върху характеристиките на CPS при дадени структура, състав и управление.

(8) Практическите резултати, постигнати в настоящата работа, са използвани в проекти на Национална Лаборатория по Компютърна вирусология с възложител "Българска Академия на Науките" и "Национален Фонд за Научни Изследвания":

1) Изследване и оценка на методите за мрежова защита от компютърни вируси (БАН – 1008001/99);

2) Разработване и използване на математически модели за оценка на разпространението на компютърни вируси (БАН – 1008002/99);

3) Изследване на влиянието на микропрограмните операции върху вирулентността на самомутиращите компютърни вируси (БАН – 1008003/99);

4) Компютърно моделиране на състав, структура и разпространение на компютърни вируси (НФНИ-И/807/99).

## ПУБЛИКАЦИИ ВЪВ ВРЪЗКА С ДИСЕРТАЦИЯТА


01. Nickolov E., *Internet and Communication Computer Virology,* Proceedings of National Conference with Foreign Participation "Development of Telecommunication Networks and Systems" TELECOM96, Oct 02-04, 1996, St. Constantine Resort, Varna, Bulgaria, pp.410-414.

02. Kodjabashev P., Nickolov E., *New Possibilities for Multi-Platform Anti-Virus Software,* Proceedings of National Conference with Foreign Participation "Development of Telecommunication Networks and Systems" TELECOM97, Oct 09-11, 1997, St. Constantine Resort, Varna, Bulgaria, pp.538-542.

03. Stanoev P., Nickolov E., *New Possibilities for Anti-Virus Protection in Compression Algorithms,* Proceedings of National Conference with Foreign Participation "Development of Telecommunication Networks and Systems" TELECOM97, Oct 09-11, 1997, St. Constantine Resort, Varna, Bulgaria, pp.549-554.

04. Stanoev P., Nickolov E., *Preventive Methods for Information Protection from Macro- Viruses,* Proceedings of National Conference with Foreign Participation "Development of Telecommunication Networks and Systems" TELECOM98, Oct 07-09, 1998, St. Constantine Resort, Varna, Bulgaria, pp.458-464.

05. Kodjabashev P., Nickolov E., *Virus Attacks on "FAT32" File System,* Proceedings of National Conference with Foreign Participation "Development of Telecommunication Networks and Systems" TELECOM98, Oct 07-09, 1998, St. Constantine Resort, Varna, Bulgaria, pp.465-470.

06. Nickolov E., *Simulation Modeling in the Information Security,* Journal Of Information Systems Security, 1998, n10, pp.343-348.

07. Nickolov E., *Information Technologies and Computer Virology,* Journal of Automatics and Informatics, 1999, n6, pp.30-39.

08. Nickolov E., Rogalska K., Balkanski D., *Information Security by Computer Stegomethods,* Proceedings of International Conference "Information Security in the 21st Century: Global Convergence", Sep 18-24, 1999, Council of Ministers Hotel, Bansko, Bulgaria, pp.65-80.

09. Nickolov E., Rogalska K., Grantcharov V., *Asymmetric Cryptography – State of Art and Prospects,* Proceedings of International Conference "Information Security in the 21st Century: Global Convergence", Sep 18-24, 1999, Council of Ministers Hotel, Bansko, Bulgaria, pp.81-100.

10. Nickolov E., Nickolova M., Grantcharov V., *SYN Attacks – A New Old Threat for the Network Security,* Proceedings of International Conference "Information Security in the 21st Century: Global Convergence", Sep 18-24, 1999, Council of Ministers Hotel, Bansko, Bulgaria, pp.175-190.
11. Nickolov E., Grantcharov V., Balkanski D., *The Y2K Problem Complicated by Some Malicious Ideas,* Proceedings of International Conference "Information Security in the 21st Century: Global Convergence", Sep 18-24, 1999, Council of Ministers Hotel, Bansko, Bulgaria, pp.417-432.
12. Nickolov E., *Analytical Optimization Studies for Computer Systems,* Computer Security Letters, 1999, n8, pp.71-76.
13. Nickolov E., *Contemporary Trends In The Development Of Information Security And Computer Virology,* An International Journal of Information and Security, Vol. 4, 2000, pp.60-72.
14. Nickolov E., *Simulation Experiments With Composition And Structure Of Computer Protection Systems,* Journal Of Advanced Communication Systems, 2000, n12, pp.211-215.
15. Nickolov E., *Information Security by Computer Protection Systems,* Proceedings of International Conference "The Inaugural Global InfoSec Summit: Building a Global Partnership", Oct 16-17, 2000, International Trade Center, Washington D.C., USA, pp.148/1-148/35.

ЛИТЕРАТУРА

1. Abad-Peiro J. L., Quisquater J. J., Quality In The Access To Information, *Research Report RZ 3017 (= 93063)*, IBM Research, April 1998, 123-136.
2. Achenberg N., Computer Virus-Antivirus Coevolution, *Communications Of The ACM*, 40, 1997, 145-153.
3. Adleman L. M., An Abstract Theory Of Computer Viruses, *Lecture Notes In Computer Science*, Springer-Verlag, Berlin, 1988, 245-288.
4. Agapow P. M., Computational Brittleness And The Evolution Of Computer Viruses, *Proceedings Of The International Conference On Evolutionary Computation*, 1141, Springer-Verlag, Berlin, 1996, 2-11.
5. Alcala J., Goodness-Of-Fit Test For Linear Models Based On Local Polynomials, *Statistics & Probability Letters*, 42, 1999, 39-46.
6. Alejandre J., Allueva A., Gonzalez J., A New Algorithm For Geometric Programming Based On The Linear Structure Of Its Dual Problem, *Mathematical And Computer Modelling*, 31, 2000, 61-78.
7. Alkhamis T., Simulated Annealing For Discrete Optimization With Estimation, *European Journal Of Operational Research*, 116, 1999, 530-544.
8. Amoroso E., *Fundamentals Of Computer Security Technology*, Prentice Hall Publishing, Englewood Cliffs, NJ, 1994, 315-360.
9. Antyufeev V., *Monte Carlo Method For Solving Inverse Problems Of Radiation Transfer*, VSP, The Netherlands, 2000, 230-357.
10. Apeland S., Aven T., Risk Based Maintenance Optimization: Foundational Issues, *Reliability Engineering And System Safety*, 67, 2000, 285-292.
11. Arsham H., Computational Geometry And Linear Programs, *International Journal Of Mathematical Algorithms*, 1, 1999, 251-262.
12. Arsham H., Input Parameters To Achieve Target Performance In Stochastic Systems: A Simulation-Based Approach, *Inverse Problems In Engineering*, 7, 1999, 363-384.
13. Arsham H., The Use Of Simulation In Discrete Event Dynamic Systems Design, *Journal Of Systems Science*, 31, 2000, 563-573.
14. Atar R., Large Deviations And Queueing Networks: Methods For Rate Function Identification, *Stochastic Processes And Their Applications*, 84, 1999, 255-296.
15. Ateniese G., Simulation Parameters For Modeling On Security Protocol Services, *Research Report RZ 3115 (= 93161)*, IBM Research, March 1999, 145-159.
16. Ateniese G., Steiner M., Tsudik G., New Multiparty Authentication Services And Key Agreement Protocols, *IEEE Journal On Selected Areas In Communications*, 18, 2000, 628-639.
17. Atkins D., Buis P., Hare C., Kelley R., Nachenberg C., Nelson A. B., Phillips P., Ritchey T., Steen W., *Internet Security Professional Reference*, New Riders Publishing, New York, NY, 1996, 130-169.
18. Averbakh I., Minimax Regret Solutions For Minimax Optimization Problems With Uncertainty, *Operations Research Letters*, 27, 2000, 57-65.
19. Azadivar F., Tompkins G., Simulation Optimization With Qualitative Variables And Structural Model Changes: A Genetic Algorithm Approach, *European Journal Of Operational Research*, 113, 1999, 169-182.
20. Bacard A., *The Computer Privacy Handbook*, Peachpit Press Inc., New York, NY, 1995, 345-410.
21. Baccelli F., Borovkov A., Mairesse J., Asymptotic Results On Infinite Tandem Queueing Networks, *Probability Theory Relative Fields*, 118, 2000, 365-405.
22. Bai Z., A Paradox In Least-Squares Estimation Of Linear Regression Models, *Statistics & Probability Letters*, 42, 1999, 167-174.
23. Balas E., Disjunctive Programming: Properties Of The Convex Hull Of Feasible Points, *Discrete Applied Mathematics*, 89, 1999, 3-44.
24. Balbas A., Jimenez P., Heras A., Duality Theory And Slackness Conditions In Multiobjective Linear Programming, *Computers & Mathematics With Applications*, 37, 1999, 101-109.
25. Bansal N., Some Characterizations Of The Normal Distribution, *Statistics & Probability Letters*, 42, 1999, 393-400.
26. Batabyal A., Beladi H., The Stability Of Stochastic Systems: The Case Of Persistence And Resilience, *Mathematical And Computer Modelling*, 30, 1999, 27-34.
27. Beirlant J., On The Impossibility Of Estimating Densities In The Extreme Tail, *Statistics & Probability Letters*, 43, 1999, 57-64.
28. Berkelaar A., Jansen B., Roos C., Terlaky T., Sensitivity Analysis For (Degenerate) Quadratic Programming, *Management Science*, 23, 1999, 256-268.

29. Bielecki T., Pliska S., Risk-Sensitive Dynamic Asset Management, *Applied Mathematical. Optimization*, 39, 1999, 337-360.
30. Bishop M., *A Security Analysis Of The NTP Protocol*, Department Of Mathematics And Computer Science, Dartmouth College, 1990, 324-375.
31. Bishop M., An Overview Of Computer Viruses In A Research Environment, *Technical Report*, Dept Of Mathematics And Computer Science, Dartmouth College, Hanover, 1992, 65-128.
32. Blondel V., Sontag E., Vidyasagar M., Willems J., (Eds.), *Open Problems In Mathematical Systems And Control Theory*, Springer-Verlag, London, 1999,122-156.
33. Bonnans J., Shapiro A., *Perturbation Analysis Of Optimization Problems*, Springer-Verlag, New York, 2000, 340-475.
34. Borgwardt K, Huhn P., A Lower Bound On The Average Number Of Pivot-Steps For Solving Linear Programs Valid For All Variants Of The Simplex-Algorithm, *Mathematical Methods Of OR*, 49, 1999, 175-210.
35. Borkowf C., A New Nonparametric Method For Variance Estimation And Confidence Interval Construction For Spearman's Rank Correlation, *Computational Statistics And Data Analysis*, 34, 2000, 219-241.
36. Bremner D., Incremental Convex Hull Algorithms Are Not Output Sensitive, *Discrete & Computational Geometry*, 21, 1999, 57-68.
37. Brinkley D. L., Schell R. R., Concepts And Terminology For Computer Security, *Information Security: An Integrated Collection Of Essays*, IEEE Computer Society Press, Los Alamitos, CA, 1995, 40-97.
38. Bueso M., Angulo J., Qian G., Alonso F., Spatial Sampling Design Based On Stochastic Complexity, *Journal Of Multivariate Analysis*, 71, 1999, 94-110.
39. Burke M., Multivariate Tests-Of-Fit And Uniform Confidence Bands Using A Weighted Bootstrap, *Statistics And Probability Letters*, 46, 1999, 13-20.
40. Campelo M, Scheimberg S., A Note On A Modified Simplex Approach For Solving Bilevel Linear Programming Problems, *European Journal Of Operational Research*, 126, 2000, 454-458.
41. Campolongo F., Braddock R., The Use Of Graph Theory In Sensitivity Analysis Of Model Output: A Second Order Screening Method, *Reliability Engineering And System Safety*, 64, 1999, 1-12.
42. Cantoni M, Marseguerra M., Zio E., Genetic Algorithms And Monte Carlo Simulation For Optimal Plant Design, *Reliability Engineering And System Safety*, 68, 2000, 291-325.
43. Carroll J. M., *Computer Security*, Butterworth-Heinemann, Newton, MA, 1995, 312-345.
44. Casado L., Garcia I., Csendes T., A New Multisection Technique In Interval Methods For Global Optimization, *Computing*, 65, 2000, 263-269.
45. Cesa-Bianchi N., Analysis Of Two Gradient-Based Algorithms For On-Line Regression, *Journal Of Computer And System Sciences*, 59, 1999, 392-411.
46. Chanas S., On The Equivalence Of Two Optimization Methods For Fuzzy Linear Programming Problems, *European Journal Of Operational Research*, 121, 1999, 56-63.
47. Chapman D. B., *Network Security Through IP Packet Filtering*, Great Circle Associates, New York, NY, 1992, 57-198.
48. Chapman D. B., Zwicky E. D., *Building Internet Firewalls*, O'Reilly & Associates Inc., Sebastopol, CA, 1995, 242-286.
49. Chelouah R., Siarry P., Tabu Search Applied To Global Optimization, *European Journal Of Operational Research*, 123, 2000, 256-270.
50. Chen P-L., Bernard E., Sen P., A Markov Chain Model Used In Analyzing Disease History Applied To A Stroke Study, *Journal Of Applied Statistics*, 26, 1999, 413-422.
51. Chen R-R., Meyn S., Value Iteration And Optimization Of Multiclass Queueing Networks, *Queueing Systems*, 32, 1999, 65-97.
52. Chess D., Grosof B., Harrison C., Levine D., Parris C., Tsudik G., Normal Agents For Mobile Computing, *IEEE Personal Communication Systems*, 2, 1995, 34-49.
53. Cheswick B., *The Design Of A Secure Internet Gateway*, AT&T Bell Laboratories, 1990, 21-245.
54. Cheswick W. R., Bellovin S. M., *Firewalls And Internet Security - Repelling The Wily Hacker*, Addison-Wesley, Reading, MA, 1994, 324-370.
55. Chipman J., Linear Restrictions, Rank Reduction, And Biased Estimation In Linear Regression, *Linear Algebra And Its Applications*, 289, 1999, 55-74.
56. Choi J., On The Stabilization Of Linear Discrete Time Systems Subject To Input Saturation, *Systems & Control Letters*, 36, 1999, 241-244.
57. Cohen F. B, Computer Viruses - Theory And Experiments, *Computers & Security*, 2, 1987, 22-35.
58. Cohen F. B, *Protection And Security On The Information Highway*, John Wiley And Sons, New York, NY, 1997, 127-265.
59. Cohen F. B, Why 32-Bit Desktops Need Virus Protection, *Datamation*, 41, 1995, 41-43.
60. Cohen F. B., *A Short Course In Computer Viruses*, John Wiley And Sons, New York, NY, 1994, 203-337.
61. Cooper F. J., Coggans C., Halvey J. K., Hughes L., Morgan L., Siyan K., Stallings W., Stephenson P., *Implementing Internet Security*, New Riders Publishing, New York, NY, 1995, 143-324.
62. Corradi G., Higher-Order Derivatives In Linear And Quadratic Programming, *Computers And Operations Research*, 28, 2000, 209-222.
63. Coyle R., Simulation By Repeated Optimization, *Journal Of The Operational Research Society*, 50, 1999, 429-438.
64. Crawford J., Gallwey T., Bias And Variance Reduction In Computer Simulation Studies, *European Journal Of Operational Research*, 124, 2000, 571-590.
65. Crema A., An Algorithm For The Multiparametric 0--1-Integer Linear Programming Problem Relative To The Constraint Matrix, *Operations Research Letters*, 27, 2000, 13-19.
66. Crema A., An Algorithm To Perform A Complete Right-Hand-Side Parametrical Analysis For A 0--1-Integer Linear Programming Problem, *European Journal Of Operational Research*, 114, 1999, 569-579.
67. Damerdji H., Nakayama M., Two-Stage Multiple-Comparison Procedures For Steady-State Simulations, *ACM Transactions On Modeling And Computer Simulations*, 9, 1999, 1-30.
68. Darlington J., Rustem B., Decreasing The Sensitivity Of Open-Loop Optimal Solutions In Decision-Making Under Uncertainty, *European Journal Of Operational Research*, 121, 1999, 343-362.
69. Davidov O., Zelen M., Exact Tests For Exponential Regression, *Journal Of Statistical Planning And Inference*, 88, 2000, 87-97.
70. De Berg M., Van Kreveld M., Overmars M., Schwarzkopf O., *Computational Geometry: Algorithms And Applications*, Springer-Verlag, Berlin, 2000, 323-490.
71. Debar H., Dacier M., Wespi A., Lampart S., A Workbench For Intrusion Detection Systems, *Technical Report RZ 6519*, IBM Zurich Research Laboratory, Säumerstrasse 4, CH-8803 Rüschlikon, Switzerland, March 1998, 121-142.
72. Dette H., On A Nonparametric Test For Linear Relationships, *Statistics And Probability Letters*, 46, 1999, 307-316.

73. Dewoody Y., Gururaj V., Martin C., Assessing Risk For Rare Events, *Journal Of Applied Statistics*, 26, 1999, 681-688.
74. Dhillon G., *Managing Information System Security*, Macmillan Press, New York, NY, 1997, 234-367.
75. Di Battista G., Tamassia R., Vismara L., Output-Sensitive Reporting Of Disjoint Paths, *Algorithmica*, 23, 1999, 302-340.
76. Ding J., Perturbation Bounds For Least Squares Problems With Equality Constraints, *Journal Of Mathematical Analysis And Applications*, 229, 1999, 631-638.
77. Dinwoodie I., Conditional Expectations In Network Traffic Estimation, *Statistics And Probability Letters*, 47, 2000, 99-103.
78. Domingo-Ferrer J., Mateo-Sanz J., Resampling For Statistical Confidentiality In Contingency Tables, *Computers And Mathematics With Applications*, 38, 1999, 13-32.
79. Domokos A., Solution Sensitivity Of Variational Inequalities, *Journal Of Mathematical Analysis And Applications*, 230, 1999, 382-389.
80. Duffuaa S., A Stochastic Programming Model For Scheduling Maintenance Personnel, *Applied Mathematical Modelling*, 23, 1999, 385-397.
81. Efromovich S., On Rate And Sharp Optimal Estimation, *Probability Theory And Related Fields*, 113, 1999, 415-419.
82. Ekel P., Approach To Decision Making In Fuzzy Environment, *Computers & Mathematics With Applications*, 37, 1999, 59-71.
83. El Azouzi R., Abbad M., Altman E., Perturbation Of Linear Quadratic Systems With Jumps Parameters And Hybrid Controls, *Mathematical Methods Of OR*, 51, 2000, 399-417.
84. El Ghaoui L., Oustry F., Lebret H., Robust Solutions To Uncertain Semi-Definite Programs, *SIAM J. Optimization*, 9, 1999, 33-52.
85. Elmaghraby S., On Criticality And Sensitivity In Activity Networks, *European Journal Of Operational Research*, 127, 2000, 220-238.
86. Eriksson K., Stable Matching In A Common Generalization Of The Marriage And Assignment Models, *Discrete Mathematics*, 217, 2000, 135-156.
87. Estrin D., Steenstrup M., Tsudik G., Protocols For Route Establishment And Packet Forwarding Across Multi-Domain Internetworks, *ACM IEEE Transactions On Networking*, February 1993, 56-70.
88. Estrin D., Tsudik G., An End-To-End Argument For Network Layer, Inter-Domain Access Controls, *Internetworking Research And Experience*, 2, 1991, 71-85.
89. Facchinei F., Fischer A., Kanzow C., On The Accurate Identification Of Active Constraints, *SIAM J. Optimization*, 9, 1999, 14-32.
90. Famoye F., Goodness Of Fit Tests For Generalized Logarithmic Series Distribution, *Computational Statistics And Data Analysis*, 33, 2000, 59-67.
91. Fang S-C., Hu C., Wang H., Wu S., Linear Programming With Fuzzy Coefficients In Constraints, *Computers & Mathematics With Applications*, 37, 1999, 63-76.
92. Fernandez J., Pelegrin B., Algorithms For The Decomposition Of A Polygon Into Convex Polygons, *European Journal Of Operational Research*, 121, 1999, 330-342.
93. Filipovic D., Invariant Manifolds For Weak Solutions To Stochastic Equations, *Probability Theory Relative Fields*, 118, 2000, 323-341.
94. Fisher D., Lipson H., Emergent Algorithms: A New Method For Enhancing Survivability In Unbounded Systems, *Proceedings Of The Hawaii International Conference On System Sciences*, ACM, 1999, 421-435.
95. Florchinger P., Risk Sensitive Nonlinear Filtering With Correlated Noises, *Applied Mathematics Letters*, 13, 1999, 97-103.
96. Ford W., *Computer Communications Security*, Prentice Hall Publishing, Englewood Cliffs, NJ, 1994, 99-425.
97. Foulds L., Wilson J., On An Assignment Problem With Side Constraints, *Computers And Industrial Engineering*, 37, 1999, 847-858.
98. Fraedrich D., Goldberg A., A Methodological Framework For The Validation Of Predictive Simulations, *European Journal Of Operational Research*, 124, 2000, 55-62.
99. Froemelt M., *Dial-Up Firewall Security*, Sun Microsystems, November 1993, 342-467.
100. Galperin E., Linear Time Algorithms For Linear Programming, *Computers & Mathematics With Applications*, 37, 1999, 199-208.
101. Gang L., Simulation-Based Goodness-Of-Fit Test For Survival Data, *Statistics And Probability Letters*, 47, 2000, 403-410.
102. Garfinkel S., Spafford G., *Practical UNIX And Internet Security*, O'Reilly & Associates Inc., Sebastopol, CA, NY, 1996, 24-345.
103. Garvey P., *Probability Methods For Cost Uncertainty Analysis: A Systems Engineering Perspective*, Marcel Dekker, 2000, 450-634.
104. Gasser M., *Building A Secure Computer System*, Van Nostrand Reinhold, New York, NY, 1988, 345-469.
105. Gelenbe E., Hernandez M., Virus Tests To Maximize Availability Of Software Systems, *Theoretical Computer Science*, 125, 1994, 131-147.
106. Genton M., De Luna X., Robust Simulation-Based Estimation, *Statistics And Probability Letters*, 48, 2000, 253-259.
107. Gerencser L., Convergence Rate Of Moments In Stochastic Approximation With Simultaneous Perturbation Gradient Approximation And Resetting, *IEEE Transactions On Automatic Control*, 44, 1999, 894-905.
108. Gil M., Perturbations Of Simple Eigenvectors Of Linear Operators, *Manuscripta Mathematica*, 100, 1999, 213-219.
109. Glaz J., Simultaneous Confidence Intervals For Multinomial Proportions, *Journal Of Statistical Planning And Inference*, 82, 1999, 251-262.
110. Glazebrook K., J. Nieo-Mora, A Linear Programming Approach To Stability, Optimization And Performance Analysis For Markovian Multiclass Queueing Networks, *Annals Of Operations Research*, 92, 1999, 1-18.
111. Goldberg L. A., Research On Computer Anti Virus Applications, *Lecture Notes In Computer Science*, 1075, 1996, 253-267.
112. Goldberg L. A., Goldberg P. W., Phillips C. A., Sorkin G. B., Constructing Computer Virus Phylogenies, *Journal Of Algorithms*, 26, 1998, 188-208.
113. Golenko-Ginzburg D., Gonik A., Papic L., Developing Cost-Optimization Production Control Model Via Simulation, *Mathematics And Computers In Simulation*, 49, 1999, 335-351.
114. Gomez G., Luz Calle M., Non-Parametric Estimation With Doubly Censored Data, *Journal Of Applied Statistics*, 26, 1999, 45-58.
115. Gross J., On Contractions In Linear Regression, *Journal Of Statistical Planning And Inference*, 74, 1999, 343-351.
116. Guenluek O., A Branch-And-Cut Algorithm For Capacitated Network Design Problems, *Mathematical Programming*, 86, 1999, 17-39.
117. Guerrero V., Selecting A Linearizing Power Transformation For Time Series, *Journal Of Applied Statistics*, 27, 1999, 185-196.
118. Guillou A., Efficient Weighted Bootstraps For The Mean, *Journal Of Statistical Planning And Inference*, 77, 1999, 11-35.
119. Guinier D., Computer Virus Identification By Neural Networks, *ACM SIGSAC Review*, 9, 1991, 49-59.

120.Gurkan G., Ozge A., Robinson S., Sample Path Solution Of Stochastic Variational Inequalities, *Mathematical Programming*, 84, 1999, 313-333.
121.Gustafsson A., F. Ekdahl, Bergman B., Conjoint Analysis: A Useful Tool In The Design Process, *Total Quality Management*, 10, 1999, 327-344.
122.Gzyl H., Tagliani A., Velasquez Y., A Comparative Study Of Some Reconstruction Methods For Linear Inverse Problems, *Mathematical And Computer Modelling*, 30, 1999, 159-167.
123.Haas P., On Simulation Output Analysis For Generalized Semi-Markov Processes, *Communication Statistical Stochastic Models*, 15, 1999, 53-80.
124.Hager W., Gowda S., Stability In The Presence Of Degeneracy And Error Estimation, *Mathematical Programming*, 85, 1999, 181-192.
125.Hallin M., Nonparametric Tests Of Independence Of Two Autoregressive Time Series Based On Autoregression Rank Scores, *Journal Of Statistical Planning And Inference*, 75, 1999, 319-330.
126.Hambridge S., Sedayao J. C., *Horses And Barn Doors: Evolution Of Corporate Guidelines For Internet Usage*, Intel Corp., November 1993, 342-490.
127.Hardwick J., Stout Q., Using Path Induction To Evaluate Sequential Allocation Procedures, *SIAM J. Scientific Computing*, 21, 1999, 67-87.
128.Harvill J., Testing Time Series Linearity Via Goodness-Of-Fit Methods, *Journal Of Statistical Planning And Inference*, 75, 1999, 331-341.
129.Hauser R., Janson P., Tsudik G., Herreweghen E. V., Molva R., Robust And Secure Password And Key Change Method, *Journal Of Computer Security*, 4, 1996, 97-111.
130.Hauser R., Przygienda A., Tsudik G., Reducing The Cost Of Security In Link State Routing, *Proceeding Of The Symposium On Network And Distributed Systems Security*, San Diego, CA, 1997, 93-99.
131.Hauser R., Zatti S., Security, Authentication And Policy Management In Open Distributed Systems, *Proceeding Of The 10th International Information Security Conference IFIP SEC'94*, Curacao, Dutch Antilles, 1994, 143-155.
132.Heide F., Vicking B., Shortest-Path Routing In Arbitrary Networks, *Journal Of Algorithms*, 31, 1999, 105-131.
133.Helsing C., Swanson M., Todd M. A., *Computer User's Guide To The Protection Of Information Resources*, NIST Special Publication, 1999, 3-450.
134.Highland H. J., A History Of Computer Viruses, *Computers And Security*, 16, 1997, 412-438.
135.Highland H. J., Procedures To Reduce The Computer Virus Threat, *Computers And Security*, 16, 1997, 439-449.
136.Hilberdink T., An Integral Formula For Taylor Coefficients Of A Class Of Analytic Functions, *Journal Of Mathematical Analysis And Applications*, 233, 1999, 266-275.
137.Ho Y., Cassandras C., Chen C., Dai L., Ordinal Optimization And Simulation, *Journal Of The Operational Research Society*, 51, 2000, 490-500.
138.Hoffman L. J. (Editor), *Rogue Programs: Viruses, Worms, And Trojan Horses*, Van Nostrand Reinhold, New York, NY, 1990, 15-528.
139.Holbrook P., Reynolds J., (Editors), *Site Security Handbook*, Internet Engineering Task Force, RFC 1244, 1991, 2-275.
140.Horvath L., On The Best Approximation For Bootstrapped Empirical Processes, *Statistics & Probability Letters*, 41, 1999, 117-122.
141.Hoyle R., *Statistical Strategies For Small Sample Research*, Thousand Oaks, CA, Sage, 1999, 342-435.
142.Hu, H., *Perturbation Analysis Of Global Error Bounds For Systems Of Linear Inequalities*, *Mathematical Programming*, 88, 2000, 277-284.
143.Huang X., Stability In Vector-Valued And Set-Valued Optimization, *Mathematical Methods Of OR*, 52, 2000, 185-193.
144.Hughes L. J. (Jnr), *Actually Useful Internet Security Techniques*, New Riders Publishing, New York, NY, 1995, 15-645.
145.Hutson A., Calculating Nonparametric Confidence Intervals For Quantile Using Fractional Order Statistics, *Journal Of Applied Statistics*, 26, 1999, 343-354.
146.Ioannides D., Exponential Inequality For Associated Random Variables, *Statistics & Probability Letters*, 42, 1999, 423-431.
147.Jacobson S., Schruben L., A Harmonic Analysis Approach To Simulation Sensitivity Analysis, *IIE Transactions*, 31, 1999, 231-243.
148.Jammalamadaka S., Changes In The General Linear Model: A Unified Approach, *Linear Algebra And Its Applications*, 289, 1999, 225-242.
149.Janson P., Molva R., Security In Open Networks And Distributed Systems, *Computer Networks And ISDN Systems*, 22, 1991, 323-346.
150.Janson P., Zatti S., *Security And Management Services In Open Networks And Distributed Systems*, Addison-Wesley, Reading, MA, 1994, 12-345.
151.Jarzemba M., Sagar B., A Parameter Tree Approach To Estimating System Sensitivities To Parameter Sets, *Reliability Engineering And System Safety*, 67, 2000, 89-102.
152.Jaulin L., Boimond J-L., Hardouin L., Estimation Of Discrete-Event Systems Using Interval Computation, *Reliable Computing*, 5, 1999, 165-173.
153.Jeong K-Y., Conceptual Frame For Development Of Optimized Simulation-Based Scheduling Systems, *Expert Systems With Applications*, 18, 2000, 299-306.
154.Jerichow A., Muller J., Pfitzmann A., Pfitzmann B., Waidner M., Real-Time Mixes: A Bandwidth-Efficient Anonymity Protocol, *IEEE Journal On Selected Areas In Communications*, 16, 1998, 495-509.
155.Kallio M, Salo S., An Interior Point Method For Solving Systems Of Linear Equations And Inequalities, *Operations Research Letters*, 27, 2000, 101-107.
156.Karjoth G., Asokan N., Gulcu C., Protecting The Computation Results Of Free-Roaming Agents, *Personal Technologies*, 2, 1998, 92-99.
157.Karjoth G., Authorization In CORBA Security, *Journal Of Computer Security*, 8, 2000, 89-108.
158.Karjoth G., Lange D. B., Oshima M., A Security Model For Aglets, *IEEE Internet Computing*, 6, 1997, 68-77.
159.Kaufman C., Perlam R., Speciner M., *Network Security: Private Communication In A Public World*, Prentice Hall Publishing, Englewood Cliffs, NJ, 1995, 34-355.
160.Kephart J. O., Sorkin G. B., Chess D. M., White S. R., Fighting Computer Viruses, *Scientific American International Edition*, 277, 1997, 56-61.
161.Kephart V., White S. R., Measuring And Modeling Computer Virus Prevalence, *Proceedings Of The IEEE Computer Society Symposium On Security And Privacy (SSP '93)*, IEEE, May 1993, 2-15.
162.Khazen M., Dubi A., A Note On Variance Reduction Methods In Monte Carlo Applications To Systems Engineering And Reliability, *Monte Carlo Methods And Applications*, 5, 1999, 345-379.
163.Kilmer R., Smithg A., Schuman L., Computing Confidence Intervals For Stochastic Simulation Using Neural Network Metamodels, *Computers & Industrial Engineering*, 36, 1999, 391-407.

164.Kim S-J, Shader B., Linear Systems With Signed Solutions, *Linear Algebra And Its Applications*, 313, 2000, 21-40.
165.Klar B., Goodness-Of-Fit Tests For Discrete Models Based On The Integrated Distribution Function, *Metrika*, 49, 1999, 53-69.
166.Klebaner F., Moderate Deviations For Randomly Perturbed Dynamical Systems, *Stochastic Processes And Their Applications*, 80, 1999, 157-176.
167.Kleijnen J., A Methodology For Fitting And Validating Metamodels In Simulation, *European Journal Of Operational Research*, 120, 1999, 14-29.
168.Kleine A., Decisions With Uncertain Alternatives, *OR Spektrum*, 21, 1999, 315-329.
169.Kochar S., On Stochastic Orderings Between Distributions And Their Sample Spacings, *Statistics & Probability Letters*, 42, 1999, 345-352.
170.Kuk A., The Use Of Approximating Models In Monte Carlo Maximum Likelihood Estimation, *Statistics And Probability Letters*, 45, 1999, 325-333.
171.Kyas O., *Internet Security*, International Thompson Publishing, New York, NY, 1997, 23-398.
172.Harn L., Lin H.-Y., Yang S., A Software Authentication System For The Prevention Of Computer Viruses, *Proceedings Of The Conference On Computer Science*, ACM Press, 1992, 447-450.
173.Lacoste G., Steiner M., SEMPER: A Security Framework For The Global Electronic Marketplace, *Comtec - The Magazine For Telecommunications Technology*, 77, September 1999, 56-63.
174.Lahiri S., Asymptotic Distribution Of The Empirical Spatial Cumulative Distribution Function Predictor And Prediction Bands Based On A Subsampling Method, *Probability Theory Relative Fields*, 114, 1999, 55-84.
175.Lamont G. B., Marmelstein R. E., Veldhuizen D. A., *A Distributed Architecture For A Self-Adaptive Computer Virus Immune System - New Ideas In Optimization*, McGraw-Hill, New York, NY, 1999, 167-183.
176.Lange D. B., Oshima M., Karjoth G., Kosaka K., Aglets: Programming Mobile Agents In Java, *Proceeding Of The 1st International Conferance On Worldwide Computing And Its Applications*, Springer-Verlag, Berlin, 1997, 253-266.
177.Lau H., A Comparison Of Procedures For Estimatintg The Parent Probability Distribution From A Given Set Of Fractiles, *European Journal Of Operational Research*, 120, 1999, 657-670.
178.Lee H., An Optimization-Based Domain Decomposition Method For A Nonlinear Problem, *Applied Mathematics And Computation*, 113, 2000, 23-42.
179.Lee L., Lau T., Ho Y., Explanation Of Goal Softening In Ordinal Optimization, *IEEE Transactions On Automatic Control*, 44, 1999, 94-98.
180.Levitin G., Lisnianski A., Joint Redundancy And Maintenance Optimization For Multistate Series-Parallel Systems, *Reliability Engineering And System Safety*, 64, 1999, 33-42.
181.Lewis A., *Option Valuation Under Stochastic Volatility With Mathematic Code*, Finance Press, Newport Beach, CA, 2000, 23-578.
182.Li W., Szidarovszky F., Notes On The Stability Of Dynamic Economic Systems, *Applied Mathematics And Computation*, 108, 1999, 85-89.
183.Lin Y., Cai G., Some Thoughts On Averaging Techniques In Stochastic Dynamics, *Probabilistic Engineering Mechanics*, 15, 2000, 7-14.
184.Lisnianski A., Levitin G., Ben-Haim H., Structure Optimization Of Multi-State System With Time Redundancy, *Reliability Engineering And System Safety*, 67, 2000, 103-112.
185.Lund U., Least Circular Distance Regression For Directional Data, *Journal Of Applied Statistics*, 26, 1999, 723-734.
186.Malik I., *Computer Hacking: Detection And Protection*, Sigma Press, New York, NY, 1996, 128-354.
187.Marazzi A., The Truncated Mean Of An Asymmetric Distribution, *Computational Statistics And Data Analysis*, 32, 1999, 79-100.
188.Mardia K., Directional Statistics And Shape Analysis, *Journal Of Applied Statistics*, 26, 1999, 949-958.
189.Martorell S., Sarlos S., Sanchez A., Serradell V., Constrained Optimization Of Test Intervals Using A Steady-State Genetic Algorithm, *Reliability Engineering And System Safety*, 67, 2000, 215-232.
190.Math P., Numerical Integration Using Markov Chains, *Monte Carlo Methods And Applications*, 5, 1999, 325-344.
191.Maul A., A Stochastic Process Applied To Sequential Parametric Analysis Of Censored Survival Data, *Journal Of Statistical Planning And Inference*, 78, 1999, 191-204.
192.Mausser H., Laguna M., Minimising The Maximum Relative Regret For Linear Programs With Interval Objective Function Coefficients, *Journal Of The Operational Research Society*, 50, 1999, 1163-1169.
193.Mavroudakis D. K., Gritzalis D., Katsikas S., Design Of A Neural Network For Recognition And Classification Of Computer Viruses, *Computers & Security*, 14, 1995, 435-448.
194.McCarthy L., *Intranet Security*, Prentice Hall Publishing, Englewood Cliffs, NJ, 1998, 12-434.
195.Menendez M., Statistical Inference For Finite Markov Chains Based On Divergences, *Statistics & Probability Letters*, 41, 1999, 9-17.
196.Michalewicz Z., Fogel D., *How To Solve It: Modern Heuristics*, Springer-Verlag, Berlin, 2000, 34-615.
197.Mistiri F., Wang A., Estimation Problems In An Input-And-Output System, *Computers & Mathematics With Applications*, 39, 2000, 29-42.
198.Mohd I., Dasril Y., Constraint Exploration Method For Quadratic Programming Problem, *Applied Mathematics And Computation*, 112, 2000, 161-170.
199.Moore J. T., Mobile Code Security Techniques, *Technical Report*, University Of Pennsylvania, Department Of Computer And Information Science, Number MS-CIS-98-28, May 1998, 25-195.
200.Mukhopadhyay N., Second-Order Properties Of A Two-Stage Fixed-Size Confidence Region For The Mean Vector Of A Multivariate Normal Distribution, *Journal Of Multivariate Analysis*, 68, 1999, 250-263.
201.Myers D., Yeh A., Generating Correlated Random Variables For A Simulation Model, *Journal Of The Operational Research Society*, 50, 1999, 183-186.
202.Nachenberg C., Computer Virus: Antivirus Coevolution, *Communications Of The Association For Computing Machinery*, 40, 1997, 46-51.
203.Nakayama M., Multiple Comparisons With The Best Using Common Random Numbers In Steady-State Simulations, *Journal Of Statistical Planning And Inference*, 85, 2000, 37-48.
204.Neumann P. G., *Computer Related Risks*, Addison-Wesley, Reading, MA, 1995, 126-457.
205.Nocedal J., Wright S., *Numerical Optimization*, Springer-Verlag, London, 1999, 43-348.
206.Obuchowska W., Minimal Representation Of Convex Regions Defined By Analytic Functions, *Journal Of Mathematical Analysis And Applications*, 246, 2000, 100-121.
207.O'Connor L., Karjoth G., Efficient Downloading And Updating Applications On Portable Devices Using Authentication Trees, *Proceedings Of The Fourth Smart Card Research And Advanced Application Conference*, Kluwer Academic Publishers, 2000, 327-343.

208.Okten G., Error Reduction Techniques In Quasi-Monte Carlo Integration, *Mathematical And Computer Modelling*, 30, 1999, 61-69.
209.Osuna-Gomez R., Beato-Moreno A., Rufian-Lizana A., Generalized Convexity In Multiobjective Programming, *Journal Of Mathematical Analysis And Applications*, 233, 1999, 205-220.
210.Ozdamar L., Demirhan M., Experiments With New Stochastic Global Optimization Search Techniques, *Computers And Operations Research*, 27, 2000, 841-865.
211.Pandey M., Extreme Quantile Estimation Using Order Statistics With Minimum Cross-Entropy Principle, *Probabilistic Engineering Mechanics*, 16, 2000, 31-42.
212.Parkinson D., Second Order Stochastic Simulation With Specific Correlation, *Advances In Engineering Software*, 30, 1999, 489-494.
213.Pearl D., Bartoszynski R., Maa J., Horn D., High-Dimensional Simulation-Based Estimation, *Mathematical And Computer Modeling*, 32, 2000, 113-124.
214.Pelletier M., An Almost Sure Central Limit Theorem For Stochastic Approximation Algorithms, *Journal Of Multivariate Analysis*, 71, 1999, 76-93.
215.Pfitzmann A., Pfitzmann B., Schunter M., Waidner M., Trusting Mobile User Devices And Security Modules, *IEEE Computer*, 30, 1997, 61-68.
216.Pfitzmann B., Schunter M., Waidner M., Secure Reactive Systems, *Research Report RZ 3206 (=93252)*, IBM Research, 2000, 25-96.
217.Pfleeger C. P., *Information Security*, Prentice Hall Publishing, Englewood Cliffs, NJ, 1999, 15-645.
218.Pfleeger C. P., *Security In Computing*, Prentice Hall Publishing, Englewood Cliffs, NJ, 1996, 34-453.
219.Pfleeger C., Cooper D., Security And Privacy: Promising Advances, *IEEE Software*, 24, 1997, 27-32.
220.Phu H., Yen N., On The Stability Of Solutions To Quadratic Programming Problems, *Mathematical Programming Online*, 2000, 27-569.
221.Polk W. T., Bassham L. E., *Anti-Virus Tools And Techniques*, Computer Security Division, National Institute Of Standards And Technology, December 1992, 45-137.
222.Pounder C., Kosten F., *Managing Data Protection Second Edition*, Butterworth-Heineman Limited, Newton, MA, 1992, 126-375.
223.Punnen A., Nair K., Constrained Balanced Optimization Problems, *Computers & Mathematics With Applications*, 37, 1999, 157-163.
**224.**Ranum M. J., *Thinking About Filewalls*, Trusted Information Systems Inc., New York, NY, 1992, 34-298.
225.Reilly T., Sensitivity Analysis For Dependent Variables, *Decision Sciences Journal*, 31, 2000, 551-572.
226.Ricotti M., Zio E., A Neural Network Approach To Sensitivity And Uncertainty Analysis, *Reliability Engineering And System Safety*, 64, 1999, 59-71.
227.Riordan J., Patterns Of Network Intrusion, *Multilateral Security In Communications*, Addison-Wesley, Reading, MA, 1999, 45-395.
228.Riordan J., Schneier B., A Certified E-Mail Protocol, *Proceeding Of The 14th Annual Computer Security Applications Conference*, ACM, 1998, 345-356.
229.Robert C., Casella G., *Monte Carlo Statistical Methods*, Springer-Verlag, New York, 1999, 15-428.
230.Rocco C., Miller S., Moreno J., Carrasquero N., Medina M., Sensitivity And Uncertainty Analysis In Optimization Programs Using An Evolutionary Approach: A Maintenance Application, *Reliability Engineering And System Safety*, 67, 2000, 249-256.
231.Romik D., Sharp Entropy Bounds For Discrete Statistical Simulation, *Statistics & Probability Letters*, 42, 1999, 219-227.
232.Roth A., Multiple Comparison Procedures For Discrete Test Statistics, *Journal Of Statistical Planning And Inference*, 82, 1999, 101-117.
233.Russell D., Gangemi G.T., *Computer Security Basics*, O'Reilly And Associates, Inc., Sebastopol, CA, 1991, 33-275.
234.Saltelli A., Chan K., Scott M., (Eds.), *Mathematical And Statistical Methods For Sensitivity Analysis*, John Wiley And Sons, New York, NY, 2000, 45-386.
235.Saltelli A., Chan K., Scott M., (Eds.), Special Issue On Sensitivity Analysis, *Computer Physics Communications*, 65, 1999, 117-131.
236.Saltelli A., Tarantola S., Chan K., A Quantitative, Model Independent Method For Global Sensitivity Analysis Of Model Output, *Technometrics*, 41, 1999, 39-56.
237.Saltelli A., Tarantola S., Chan K., A Role For Sensitivity Analysis In Presenting The Results From MCDA Studies To Decision Makers, *Journal Of Multi-Criteria Decision Analysis*, 8, 1999, 139-145.
238.Schneier B., *Applied Cryptograhpy*, John Wiley And Sons, New York, NY, 1996, 34-468.
239.Schneier B., Cryptography, Security And The Future, *Communications Of The Association For Computing Machinery*, 40, 1997, 154-169.
240.Schobel A., Solving Restricted Line Location Problems Via A Dual Interpretation, *Discrete Applied Mathematics*, 93, 1999, 109-125.
241.Seltzer M. I., Endo Y., Small C., Smith K. A., Dealing With Disaster: Surviving Misbehaved Kernel Extensions, *Proceedings Of The 2nd Symposium On Operating Systems Design And Implementation*, 1996, 213-227.
242.Sexton R., Dorsey R., Johnson J., Optimization Of Neural Networks: A Comparative Analysis Of The Genetic Algorithm And Simulated Annealing, *European Journal Operational Research*, 114, 1999, 589-601.
243.Shao S., Percy P., Yip C., Rates Of Convergence Of Adaptive Step-Size Of Stochastic Approximation Algorithms, *Journal Of Mathematical Analysis And Applications*, 244, 2000, 333-347.
244.Sherman M., Efficiency And Robustness In Subsampling For Dependent Data, *Journal Of Statistical Planning And Inference*, 7, 1999, 133-146.
245.Shi L., Olafsson S., Nested Partitions Method For Global Optimization, *Operations Research*, 48, 2000, 390-407.
246.Shih N-H., The Sensitivity Analysis Of Binary Networks Via Simulation, *European Journal Of Operational Research*, 114, 1999, 602-609.
247.Shoup V., On Formal Models For Secure Key Exchange, *Research Report RZ 3120 (=93166)*, IBM Research, 1999, 34-156.
248.Shoup V., On The Security Of A Practical Identification Scheme, *Journal Of Cryptology*, 12, 1996, 247-260.
249.Sibert O., Porras P. A., Lindell R., The Intel 80x86 Processor Architecture: Pitfalls For Secure Systems, Proceedings 1995 IEEE Symposium On Security And Privacy, Oakland, CA, 1995, 211-222.
250.Siyan K., Hare C., *Internet Firewalls And Network Security*, New Riders Publishing, New York, NY, 1995, 35-365.
251.Slade R., *Computer Viruses*, Springer-Verlag, Berlin, 1996, 12-286.
252.Soh B. C., Dillon T. S., Countym P., Quantitative Risk Assessment Of Computer Virus Attacks On Computer Networks, *Computer Networks And ISDN Systems*, 27, 1995, 1447-1456.
253.Song W., A Three-Class Variance Swapping Technique For Simulation Experiments, *Operations Research Letters*, 23, 1999, 63-70.

254.Soofi E., A Framework For Measuring The Importance Of Variables With Applications To Management Research And Decision Models, *Decision Sciences Journal*, 31, 2000, 595-625.
255.Stadje W., Joint Distributions Of The Numbers Of Visits For Finite-State Markov Chains, *Journal Of Multivariate Analysis*, 70, 1999, 157-176.
256.Stallings W., *Network And Internetwork Security: Principles And Practice*, Prentice Hall Publishing, Englewood Cliffs, NJ, 1995, 34-610.
257.Steiner M., Tsudik G., Waidner M., Key Agreement In Dynamic Peer Groups, *IEEE Transactions On Parallel And Distributed Systems*, 11, 2000, 769-780,
258.Sterling B., *The Hacker Crackdown*, Viking Press, New York, NY, 1993, 124-296.
259.Stinson D. R., *Cryptography Theory And Practice*, CRC Press, New York, NY, 1995, 26-456.
260.Sui Y, Two Optimization Methods For Complicated Structures With Multiparameter Elements Based On Generalized Condensation, *Structural Optimization*, 17, 1999, 226-232.
261.Summers R., *Secure Computing*, McGraw-Hill, New York, NY, 1997, 35-423.
262.Sun H., Testing For Trends In Correlated Data, *Statistics & Probability Letters*, 41, 1999, 87-95.
263.Sung S., Weak Law Of Large Numbers For Arrays Of Random Variables, *Statistics & Probability Letters*, 42, 1999, 293-298.
264.Taa A., Optimality Conditions For Vector Mathematical Programming Via A Theorem Of The Alternative, *Journal Of Mathematical Analysis And Applications*, 23, 1999, 233-245.
265.Thimbleby H., Anderson S., Cairns P., A Framework For Modelling Trojans And Computer Virus Infections, *The Computer Journal*, 41, 1998, 444-458.
266.Traub J., Werschulz A., *Complexity And Information*, Cambridge University Press, 1999, 23-390.
267.Treese G. W., Wolman A., *Through The Firewall, And Other Application Relays, Technical Report CRL 93 10*, Digital Equipment Research Laboratory, May 1993, 124-257.
268.Tsinias J., The Concept Of 'Exponential Input To State Stability' For Stochastic Systems And Applications To Feedback Stabilization, *Systems & Control Letters*, 36, 1999, 221-229.
269.Tunali S., Batmaz I., Dealing With The Least Squares Regression Assumptions In Simulation Metamodeling, *Computers & Industrial Engineering*, 38, 2000, 307-320.
270.Usabel M., Practical Approximations For Multivariate Characteristics Of Risk Processes, *Insurance Mathematics And Economics*, 25, 1999, 397-413.
271.Van Hoesel S., On The Complexity Of Postoptimality Analysis Of 0/1 Programs, *Discrete Applied Mathematics*, 91, 1999, 251-263.
272.Van Tu T., Optimization Over The Efficient Set Of A Parametric Multiple Objective Linear Programming, *European Journal Of Operational Research*, 122, 2000, 570-583.
273.Vassiliou P., The Perturbed Nonhomogeneous Markov Systems, *Linear Algebra And its Applications*, 289, 1999, 319-332.
274.Vaurio J, Identification Of Process And Distribution Characteristics By Testing Monotonic And Non-Monotonic Trends In Failure Intensities And Hazard Rates, *Reliability Engineering & System Safety*, 64, 1999, 345-357.
275.Wack J. P., *Establishing A Computer Security Incident Response Capability (CSIRC)*, NIST Special Publication 800-3, 1991, 34-121.
276.Wang H., Solution Of The System Of Linear Algebraic Equations By Decreasing Dimension, *Applied Mathematics And Computation*, 109, 2000, 51-57.
277.Wang M., Kuo Y., A Perturbation Method For Solving Linear Semi-Infinite Programming Problems, *Computers & Mathematics With Applications*, 37, 1999, 181-198.
278.Wang P., Sequencing And Scheduling Customers For A Stochastic Server, *European Journal Of Operational Research*, 119, 1999, 729-738.
279.Wang T., Wu K., A Parameter Set Design Procedure For The Simulated Annealing Algorithm Under The Computational Time Constraint, *Computers & Operations Research*, 26, 1999, 665-678.
280.Wludyka P., Nelson P., Two Non-Parametric, Analysis-Of-Means-Type Tests For Homogeneity Of Variances, *Journal Of Applied Statistics*, 26, 1999, 243-256.
281.Xia Y., Liu B., Wang S., Lai K., A Model For Portfolio Selection With Order Of Expected Returns, *Computers And Operation Research*, 27, 2000, 409-422.
282.Yamamura K., Finding All Solutions Of Nonlinear Equations Using Linear Combinations Of Functions, *Reliable Computing*, 6, 2000, 105-113.
283.Yap C., *Fundamental Problems Of Algorithmic Algebra*, Oxford University Press, 2000, 43-396.
284.Yen H-C., Integer Linear Programming And The Analysis Of Some Petri Net Problems, *Theory Of Computing Systems*, 32, 1999, 467-485.
285.Yin B., Zhou Y., Xi H., Sun D., Sensitivity Formulas Of Performance In Two-Server Cyclic Queueing Networks With Phase-Type Distributed Service Times, *International Transactions In Operational Research*, 6, 1999, 649-663.
286.Yoo J., Hajela P., Immune Network Simulations In Multicriterion Design, *Structural Optimization*, 18, 1999, 85-94.
287.Zarepour M., Bootstrapping Point Processes With Some Applications, *Stochastic Processes And Their Applications*, 84, 1999, 81-90.
288.Zhang J., Xu X., An Efficient Evolutionary Programming Algorithm, *Computers & Operations Research*, 26, 1999, 645-663.
289.Zhang S., New Variants Of Finite Criss-Cross Pivot Algorithms For Linear Programming, *European Journal Of Operational Research*, 116, 1999, 607-614.
290.Zheng A., Robust Stability Analysis Of Constrained Model Predictive Control, *Journal Of Process Control*, 9, 1999, 271-278.
291.Zimmerman P. R., *PGP: Source Code And Internals*, The MIT Press, Boston, MA, 1995, 45-375.

Разпределение по години:

| | | |
|---|---|---|
| 2000_065_22% | 1999_143_48%_ | 1998_009_03% |
| 1997_015_05% | 1996_011_04% | 1995_017_06% |
| 1994_007_03% | 1993_006_02% | 1992_006_02% |
| 1991_006_02% | 1990_003_01% | 1988_002_01% |
| 1987_001_01% | | |