ALGORITHM OF DECISION SUPPORT IN EMERGENCY

Oleksandr Kuzomin, Valentina Davidenko, Illya Klymov

Abstract: Constant increase of human population result in more and more people living in emergency dangerous regions. In order to protect them from possible emergencies we need effective solution for decision taking in case of emergencies, because lack of time for taking decision and possible lack of data. One among possible methods of taking such decisions is shown in this article.

Keywords: emergency ,decision taking, time management

ACM Classification Keywords: H.3.4 Systems and Software

Introduction

Rapid growth of population on our planet inevitably leads to more and more people located on the territories which are affected by that or another dangers. According to United Nations statistics, more than 6% of world population is living on territories, annually suffering from emergency situations. Taking that into account, effective prediction, control and prevention measures for emergencies become extremely actual.

The task of complex development for preventing emergency situations is quite complex task of multi criteria optimization. For person, taking decisions, a problems of human risks and material losses minimization and also money losses are very actual. Taking decision in case of very limited time also leads to risk of errors and human factor impact.

Attemps to apply these or another automated complexes of providing solutions in emergency situations often lead to a problem of internal system complex representation (complex of links “operator-available resources-current situation-complex of measures to prevent emergency”), where change of one parameter can greatly affect selection of technical and human means.

So, the task of context-dependent measures complex development in emergency situations becomes important. In fact, the primary aim of this article is research of possible appliances of context-depended solutions in such cases.

Problem statement

As a parameter for our system there is a current situation Sit, which could be represented as set of substances' features and interconnections between them.

$$Sit = cont_k = \{A, L\},$$

where A = $\{a_1, \dots, a_m\}$ – set of environment substances and, m – their amount; $L = \{l_1, \dots, l_s\} = \{(a_i, a_j, c)\}$- is a set of connections between pairs of environment substances. $i = \overline{1,m};\ j = \overline{1,m};\ c \in \{0,1\}$.

In role of substances in current context depending on system task may appear different nature objects (wood array, sea, mountain hill), natural phenomenons (air stream, underwater streams), subjects (rescue rangers, scientific workers, researchers), material wealth (buildings, strategical military objects).

Each substance of context have a set of properties:

$$\forall a_i : P_{a_i} = \{p_{a_{i_1}}, \dots, p_{a_{i_{k_i}}}\},$$

where P_{a_i} – is a subset of properties for substance a_i. k_i – is a count of properties of substance a_i. Examples of substance “air flow” properties can be “velocity”, “direction” and so on.

As another substance of the system we will present a set of solutions D = {d_1, …, d_n}, which describes all available means for preventing and protecting from emergency.

As a data storage for taken earlier decisions in that or another context we will use a knowledge base (KB). Format of data storage will be triple system “context-solution-effectiveness”. Coefficient of effectiveness q is calculated depending on material and non-material losses in emergency context based on taken decision. In fact minimization of that parameter is the main criteria of system effectiveness.

Algorithm of optimum decision finding

For finding optimal decision in unknown context (we specify context as unknown if it isn’t found in our KB) let’s create a conversion function: $d_k' = F(cont_k)$, where $d_k' \in D'$ - set of new obtained from KB solutions; $F(x)$ – function of solution taking for context x. Principal scheme of that method implementation is presented on figure 1.

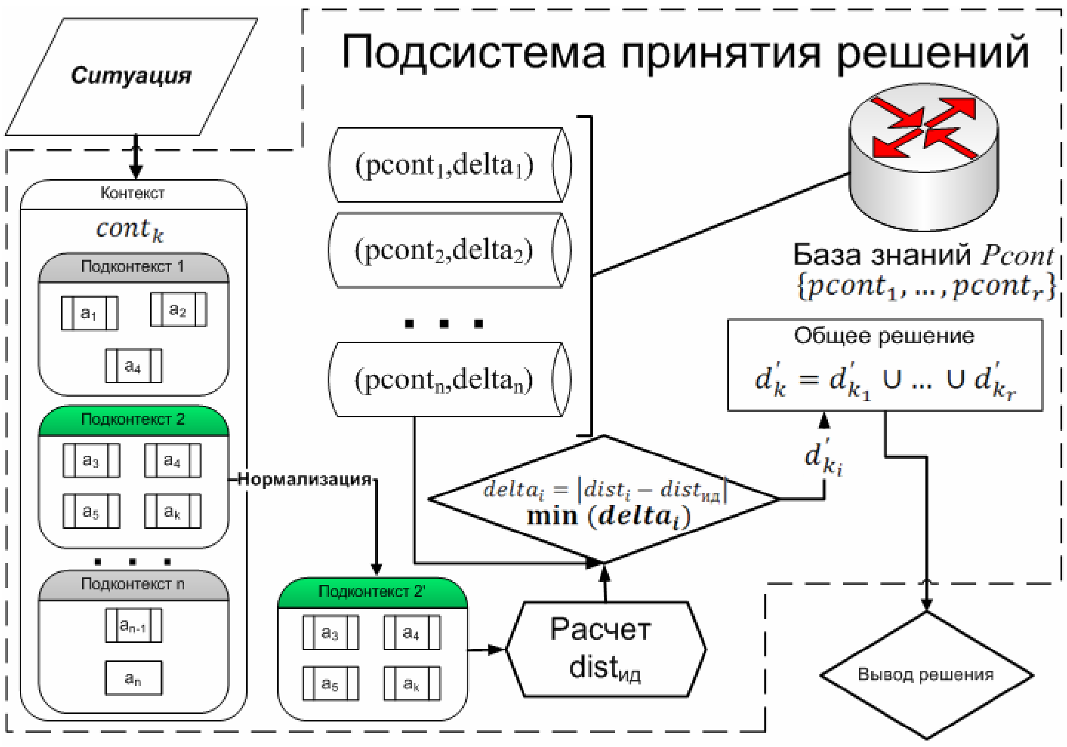

Fig. 1. Principal scheme of algorithm.

$F(x)$ execution algorithm:

1. Separate the environment context to sub-contexts. Separation will be performed according to common meaning of that situation. Sub-contexts are obtained both in current contexts and contexts, extracted from KB.
2. Create a set of sub-contexts $Pcont = \{pcont_1, \dots, pcont_r\}$, where $pcont_i = \{a_1, \dots, a_{n_z}, l_1, \dots, l_{s_z}\}$, where $\forall i, k: pcont_i \cap pcont_k \geq \emptyset$; r – amount of sub-contexts;

General solution for all context is composed from solutions for separate sub-contexts $d_k^* = d_{k_1}^* \cup \dots \cup d_{k_r}^*$, where $d_{k_i}^*$ - solution for sub-context $pcont_i$.

3. For each sub-context perform a properties normalization using formula: $x_n = \frac{x}{x_{max}}$, where x – current value; x_{max} – maximal possible value of that property; x_n – normalized value. A parameter q in KB is also normalized.
4. Introduce a virtual situation, where in presented sub-context losses will be minimal (q=0). For each sub-context of new description we find in KB nearest contexts, which are close to ideal situations.

Let's introduce to our system parameter [A] – vector of coefficients attached to independent variables, which define logical distance from ideal situation to nearest context in KB with optimal solution. Independent variables are the values of substance properties for current sub-context and parameter "q" for current solution (taken from KB).

Define [P^q] – as a vector of properties for sub-context substance, where last element is parameter *q*, which describes losses of current sub-context in KB; we receive it using conversion of properties set and adding element *q* to them.

Include set $Dist = \{dist_1, \dots, dist_N\}$, where $dist_i = [A] * [P_i^q]$, N – total amount of records in KB.

Criteria of how current sub-context is close to current (k) sub-context is $delta_k$:

$delta_i = |dist_i - dist_{id}|$, where $dist_{id}$ – is characteristics of ideal state. We add a set Delta = { $delta_1$, …, $delta_z$}, where z – amount of parameters $delta$ for current sub-context.

After creating such sets we normalize their values.

After finding $delta$ for all sub-context we find minimal parameter $delta_{min}$ and add it to the set $Delta_{min} = \{delta_{min_1}, \dots, delta_{min_r}\}$, where r – amount of sub-contexts. In that case we store solution, where we reach $delta_{min}$, and add it to general solution.

5. We assume that there is a predefined information about structure information about solution interactions: $G = \{(g_i, g_j, c)\}$, where g_i, g_j – different solutions, c $\in$ {-1,0,1,2}, where

   if c = -1 – events g_i and g_j are not compatible;

   if c = 0 – events g_i and g_j can be performed parallel;

   if c = 1 – event g_i must follow before event g_j;

   if c = 2 – event g_j must follow before event g_i.

According to this rules we perform a pair check of all proposed events (in term "event" we describe a possible mean to prevent emergency) and according to results of such comparison we perform a topological sort in order to receive a schedule of performing actions.

Conflicts resolution and system learning

Provided by F(x) solution may content conflicting data. For example system may recommend to perform a forced avalanche launch, and do not allow it in another sub-solution. We will use a set $Delta_{min}$ to eliminate such conflicts.

Selecting from composed set $d_k^{'}$ for context $cont_k$ we choose positive oriented solutions of same type and sum parameters of same type $delta_{min}$ (let call it k_{Σ}^{+}), so we perform the same for negative oriented recommendations (receive k_{Σ}^{-}). So, if k_{Σ}^{+}>k_{Σ}^{-}, we save a positive solution, if k_{Σ}^{+}<k_{Σ}^{-} , we save only negative oriented solution. If k_{Σ}^{+}=k_{Σ}^{-} we need an expert opinion.

System learning is performed by finding optimal system parameters in matrix [A]. This process performs by using classical scheme by learning with teacher, where we use a teacher to correct provided data.

So we learn system as follows:

1. We provide a context to system input.
2. System provides a current estimation.
3. Expert analyzes provided solution and performs corrections. If there are corrections we perform correction of matrix [A] elements, in such way that $[A] * [P_{optimal}^{q}]$ was "arranged left" to $[A] * [P_{expert}^{q}]$.

Authors' Information

Oleksandr Kuzomin – Head of Innovation marketing department, professor in informatics dept.; Kharkiv National University of Radio electronics, e-mail: kuzy@kture.kharkov.ua

Valentina Davidenko – Postgraduate, National Aero Space University in the name of N.E. Jukovskiy;
e-mail: davidenko.valentina@gmail.com

Illya Klymov – Postgraduate, Informatics dept.; Kharkiv National University of Radio Electronics;
e-mail: ilia.klimov@gmail.com