Modeling Adaptive Behavior with Conceptual Spaces

Michael Cebulla

Technische Universität Berlin, Fakultät für Elektrotechnik und Informatik,
Institut für Softwaretechnik und Theoretische Informatik,
Franklinstr. 28/29, 10587 Berlin
mce@cs.tu-berlin.de

Abstract. We discuss a membrane-based calculus for the combination of conceptual spaces during runtime. Since our goal is to support emergent properties of behavior (and due to the fact that it is not possible to define a complete calculus for all situations) we introduce the notion of self-modification. Terms from situational description can evolve according to simple rules thus providing various possibilities for reactions.

1 Introduction

In this paper some of the problems which are connected with context-dependent behavior are developed as a general problem of combining conceptual spaces. Examples from the field of medical workflows are given. The approach which is discussed in this paper relies on the notion of conceptual integration which was developed in the field of cognitive semantics [2]. This integration can be simulated by the rule-based integration of ontologies.

The framework described in this paper partly follows previous proposals for the formalization of contextual reasoning. While classifications [1] heavily rely on universal algebra or category theory [3] we focus on an operational treatment using membrane computing [4].

In this paper we make an attempt to bridge the gap between highly reactive behavior during runtime and the need for highly abstract and meaningful concepts for context-awareness. Especially we propose to integrate highly abstract forms of common sense reasoning (as proposed by [1]) with membrane computing (as proposed by [4]) in order to support a way of runtime reasoning whose robustness is comparable to human reasoning. By this proposal we extend previous suggestions concerning high-level and intuitive specifications (cf. [5]). Especially we propose to exploit common sense reasoning for the robustness of context-aware behavior in distributed systems.

2 Context and Behavior

Simplified models of medical workflows are employed in this paper as examples for the treatment of adaptive behavior.

Example 1 (Intubation: A Medical Workflow) *The activity of intubation is considered with represents a specific part of a medical operation. Although there is certainly a definition of the process (i.e. a pattern) the exact shape of the final activity highly* *depends on the context in which this pattern is activated. In this paper we propose to represent the definition of the process (the pattern) as well as the situation as input spaces (in the sense of [2]). We develop an emergent calculus which establishes links between these spaces.* ♢

The Intubation Space. The conceptual space containing the process of intubation contains a constraint-based description of this process (cf. Figure 2). The important actions are described with their causal relationships as well as constraints which have to hold in certain states. Especially three subtasks can be identified *(Preparation, Laryngoscopy and Introduction)* which have different relevance values for the overall process. Since the intubation space contains a pattern which is described in this space there are many variables which have to be bound to actual values from a specific situation. For instance, agents are represented by *roles* which have to be bound to real agents taken from another conceptual space. In a similar way constraints which are specified over objects or states are applied to elements from other spaces.

The Situation Space. While the intubation space contains *roles* for the agents which are responsible for certain actions the situation space is populated by (entities representing) real agents and resources. In addition in this conceptual space specific relations and circumstances can be described which are of informal nature but which heavily influence the shape of the resulting process. As an example a relation of informal hierarchy is given which may hold between an experienced nurse and a less experienced anesthesist.

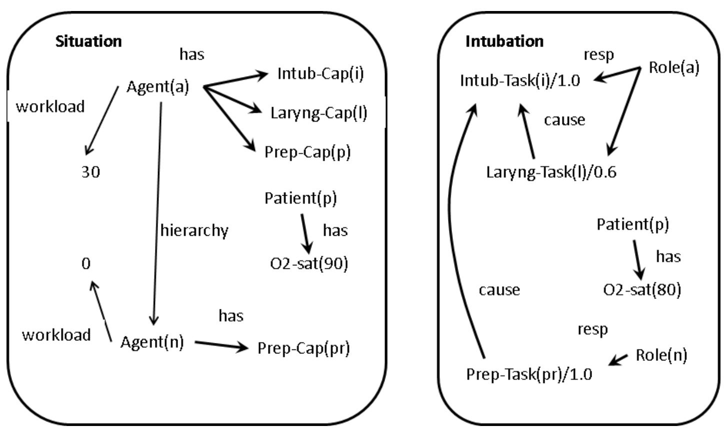

Fig. 1. Conceptual Spaces

Cross-Space Mapping. The combination of conceptual spaces is triggered by cross-space mappings. Cross-space mappings are enabled by morphisms between ontologies. Morphisms represent background knowledge for combining conceptual spaces. In our example relevant morphisms are:

```
mapping intsit from Intubation to Situation
   sort Intubation-Task Intubation-Capability
```

As we will see the background knowledge is used to establish infomorphisms between the conceptual spaces. As we will see there are multiple possibilities to establish these infomorphisms. One of our main points consist in the claim that the adaptive or self-configuring capabilities of complex systems (like human teams in the operation theatre) can be simulated by an adequate selection of the best possibility.

Generally the process of blending results in the creation of a blended space. Due to space restrictions we concentrate on the establishment of vital relations in this paper.

3 Operational Treatment of Vital Relations

Under operational aspects we represent conceptual spaces as P-systems P_{CS}. Basically a conceptual space is enclosed by a membrane. These entities which are contained in the space are mapped to components of P-systems (cf. [4]). While concepts are mapped to molecules, individuals and relations are mapped to labeled membranes.

Definition 1 (P-System P_{CS}) *The P-System for the representation of conceptual spaces P_{CS} is defined by the tuple $\langle V, L, \mu, w_i, Rn_i \rangle$ where V is the terminology of the classifications (containing concept names) and the label algebra L (containing individuals, situations and the concatenation operator ","). μ is the structure of membranes containing the multi-fuzzy sets w_i. The rules contained in Rn_i are discussed below.* □

One of our central goals is to support self-organizational capabilities in the dynamic composition of conceptual spaces. This is especially due to the fact that it is impossible to foresee every possible combination of situations. Since we do not want to define a uniform rigid calculus which is restricted to a certain set of known combinations we take the opposite approach which promises a more flexible solution. This means that we allow the terms to evolve in a solution and to look for possible combinations by themselves. This decentralized approach is robust against local evolutions and to unforeseen changes.

We proceed in two steps. Firstly we have to map context descriptions to membrane structures. This can be easily done by mapping individuals and situations to labeled membranes and concepts to molecules floating in a solution. In the same way we have to represent ontology morphisms by membrane structures. In a second step we give the rules for the evolution of these structures and for the establishment of valid combinations of contexts.

Airlock Rules. In our membrane-based approach molecules are enclosed by membranes. In order to make reactions possible however they have be able to leave their membranes. This is defined by the airlock rule. We introduce an extended version (EAL) which enables molecules to cross multiple membranes.

$$\begin{array}{ll} \text{(AL)} & [_a C_1]_a \leftrightharpoons C_1 \lhd_{\langle a \rangle} [_a]_a \\ \text{(EAL)} & [_b C_1 \lhd_{\langle L,a \rangle} [_a]_a]_b \leftrightharpoons C_1 \lhd_{\langle L,a,b \rangle} [_b[_a]_a]_b \end{array}$$

Intuitively we enable the molecules to travel through the membrane structure keeping track of the membranes they crossed in a list which is an annotation of the airlock-operator.

Interaction. The main goal is to find and encourage possible interactions. Especially molecules from situations should react with molecules from morphisms. Such reactions are only possible because both situations and morphisms evolve according to the airlock rules. The rule for interaction can be given as follows.

$$\text{(INT)} \quad C_1^+ \lhd_{\langle src,mor\rangle} [_{mor}[_{src}]_{src}]_{mor}, C_1^- \lhd_{\langle a,s1\rangle} [_{s1}[_{a}]_{a}]_{s1} \rightarrow \\ [_{\langle mor,s1\rangle}[_{\langle src,s1,a\rangle}C_1]_{\langle src,s1,a\rangle}]_{\langle mor,s1\rangle}[_{\langle s1,mor\rangle}[_{\langle a,mor\rangle}C_1]_{\langle a,mor\rangle}]_{\langle s1,mor\rangle}$$

Intuitively the reaction between the molecules is recorded by the labels. Thus the labels of the morphism are added to the labels of the situation. In the same way the labels of the morphism are extended. Note that we only treat the matching of the source ontology of the morphism. We presume that the molecules of the morphism are charged negatively while the molecules of the situation are charged positively. Note that there can be many different reactions between situations and morphisms because many copies of the structures are floating in the solution.

Completing the Infomorphism. An ontology morphism is completely bound when two molecules from two context description have been bound to its source and target ports. Since the knowledge about the creation of the bindings is contained in the labels the information is present which which completes an infomorphism (i.e. the relation between the individuals which is contravariant to the original ontology morphism).

Compression. Elements from the input spaces which are connected by a vital relation (i.e. infomorphisms) are projected into the blend. The resulting individuals which are created in the blend can be considered a tuple-valued individuals which establish a connection between the original tokens (or es equivalence classes). We cannot deepen these issues due to space restrictions.

4 Outlook

We discussed a membrane-based calculus for the creation of infomorphisms between conceptual spaces during runtime. We consider this line of research as a contribution to the exploration of adaptive and context-aware behavior in distributed systems. The treatment of infomorphisms is the strategic foundation for the integration of more advanced formal constructs from common sense reasoning.

References

1. J. Barwise and J. Seligman. *Information Flow: the logic of distributed Systems.* Cambridge Univ. Press, 1997.
2. G. Fauconnier and M. Turner. *The Way we Think.* Basic Books, 2002.
3. J. Goguen. Information integration in institutions. In *Memorial Volume for Jon Barwise.* Indiana University Press, Indiana, 2005.
4. G. Păun. *Membrane Computing. An Introduction.* Springer, Berlin, 2002.
5. P. Pepper, M. Cebulla, K. Didrich, and W. Grieskamp. From program languages to software languages. *The Journal of Systems and Software*, 60, 2002.