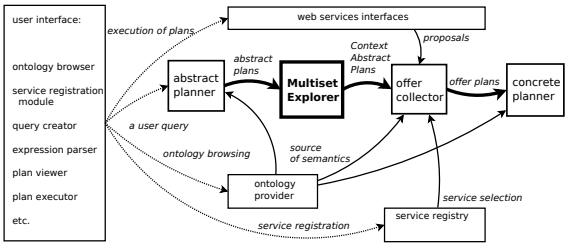
Generating CA-Plans from Multisets of Services

Łukasz Mikulski[1], Artur Niewiadomski[2], Marcin Piątkowski[1], Sebastian Smyczyński[3]

[1] Nicolaus Copernicus University,
{lukasz.mikulski, marcin.piatkowski}@mat.umk.pl
[2] Siedlce University, artur.niewiadomski@uph.edu.pl
[3] Simplito Computer Science Lab, s.smyczynski@simplito.com

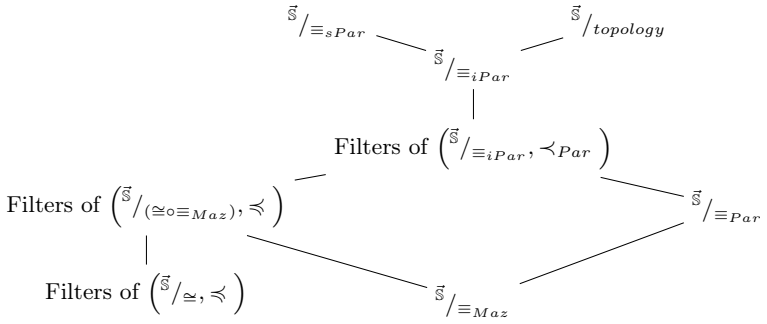
The main idea of solving WSCP utilised by PlanICS(see [3]) is to divide the composition process into several stages. The first phase, called *abstract planning*, deals with an *ontology* which contains a hierarchy of *classes* describing sets of real-world services and processed object types. Our abstract planners find multisets of service types that potentially satisfy a *user query*. Still, each equivalence class defined by a multiset can be viewed as the union of finer equivalence classes defined by partial orders, in which the plans differ only in the ordering of context independent services. Finding all of such classes is the task of Multiset Explorer - a module of PlanICS presented here.



The realizations of web service composition is a transformation sequence of services together with sets of affected objects (the arguments of those services). In contrast to concrete plan, we abstract from objects attributes. We also abstract from concrete object names (defining an equivalence relation $\cong$). We treat two sequences as indistinguishable if they differ only in types of arguments which are in inheritance relation (we build a partial order $\preccurlyeq$ based on inheritance relation and utilize the filters over $\preccurlyeq$) or are equivalent in Mazurkiewicz sense (see [2]), which we denote by $\equiv_{Maz}$. However, we distinguish between two sequences that match produced objects with the expected ones (specified by user query) differently.

The diagram presented below shows relationships between classes of transformation sequences obtained by dividing the set of all potential ones that starts with the set of initial objects specified in the user query. We denote this set by $\vec{\mathbb{S}}$. At the bottom of this diagram individual transformation sequences ($\vec{\mathbb{S}}/_{\mathbb{I}}$), can be seen. Looking at the top this diagram, we define three equivalence relations based on Parikh equivalence of services utilized in the transformation sequence. Namely, they are $\equiv_{sPar}$ which looks only on names of services (as in abstract plan), $\equiv_{Par}$ which takes into account names of the attributes and lying in be-

tween $\equiv_{iPar}$ which abstracts from the names and types of objects in favor of the inheritance relation. In our solution we cut classes of $\equiv_{sPar}$ into classes of $\equiv_{iPar}$ using the notion of relational structures (see [5]) and based on them equivalence relation $\equiv_{topology}$.



The main goal of the presented procedure is to browse all transformation sequences satisfying a given user query with the same Parikh vector of service specifications without duplicating indistinguishable ones. As an input we take the ontology, the user query in the form of two sets of objects, and an arbitrary multiset of service names that identifies single equivalence class $\equiv_{sPar}$. We start from fixing the names of objects originating form the user query initial world or produced by the considered services. After that, we distribute them between inputs of the services to obtain all possible topologies and compute maximal (in the sense of $\preccurlyeq$) possible types of utilized objects. In the next step, we match the obtained possibilities with the user query expected world, considering all valid matchings. The last step is browsing all traces (in Mazurkiewicz sense) based on the multisets of context services from $\vec{\mathbb{S}}/_{\equiv_{iPar}}$. This phase of the algorithm is based on the approach presented in [4] adapted to the specification in the form of a relational structure.

The preliminary experimental results are very promising. We used Z3 SMT-solver together with Abstract Planner (AP). We browse all solutions equivalent in the sense of $\equiv_{sPar}$ with the one reported by AP. In the case of the shortest plans we are able to validate their uniqueness significantly faster than the procedure of their generation. For longer plans, we are as fast as Z3+AP reporting from 1.5 to 5 times more plans.

Acknowledgements This research was supported by the National Science Center under the grants No.2011/01/B/ST6/01477 and No.2013/09/D/ST6/03928.

References

1. L. de Moura and N. Bjørner. Z3: An efficient SMT solver. LNCS 4963:337-340, Springer, 2008.
2. V. Diekert and G. Rozenberg, editors. *The Book of Traces.* World Scientific, 1995.
3. D. Doliwa et al. PlanICS - a web service composition toolset. *Fund. Inf.*, 112(1):47-71, 2011.
4. Ł. Mikulski et al. Algorithmics of posets generated by words over partially commutative alphabets (extended). *Scientific Annals of Comp. Sci.*, 23(2):229-249, 2013.
5. R. Janicki et al. Causal structures for general concurrent behaviours. In *CS&P'13*, pp. 193-205.