Modelling Search Strategies in Rules2CP

François Fages and Julien Martin

INRIA Rocquencourt, BP105, 78153 Le Chesnay Cedex, France
http://contraintes.inria.fr

In this abstract, we present a rule-based modelling language for constraint programming, called Rules2CP [1], and a library PKML for modelling packing problems. Unlike other modelling languages, Rules2CP adopts a single knowledge representation paradigm based on logical rules without recursion, and a restricted set of data structures based on records and enumerated lists given with iterators. We show that this is sufficient to model constraint satisfaction problems together with search strategies, where *search trees* are expressed by *logical formulae*, and *heuristic choice criteria* are defined by *preference orderings on variables and formulae*. Rules2CP statements are compiled to constraint programs over finite domains (currently SICStus-prolog and soon Choco-Java) by term rewriting and partial evaluation.

The Packing Knowledge Modelling Language (PKML) is a Rules2CP library developed in the European project Net-WMS for dealing with real size non-pure bin packing problems in logistics and automotive industry. PKML refers to shapes in $\mathbb{Z}^K$. A *point* in this space is represented by the list of its K integer coordinates. A *shape* is a *rigid assembly of boxes*, represented by a record. A *box* is an orthotope in $\mathbb{Z}^K$. An *object* , such as a bin or an item, is a record containing one attribute `shapes` for the list of its *alternative shapes*, one *origin* point, and some optional attributes such as weight, virtual reality representations or others. The alternative shapes of an object may be the discrete rotations of a basic shape, or different object shapes in a configuration problem. The end in one dimension and the volume of an object with alternative shapes are defined with reified constraints.

PKML uses Allen's interval relations in one dimension, and the topological relations of the Region Connection Calculus in higher-dimensions, to express placement constraints. Pure *bin packing problems* can be defined as follows:

```
non_overlapping(Items,Dims) --> forall(O1,Item, forall(O2,Items,
        uid(O1)<uid(O2) implies notoverlap(O1,O2,Dims))).
containmentAE(Items,Bins,Dims) -->
        forall(I,Items, exists(B,Bins, contains_rcc(B,I,Dims))).
bin_packing(Items,Bins,Dims) --> containmentAE(Items,Bins,Dims)
        and non_overlapping(Items,Dims) and labeling(Items).
```

Other rules about weights, stability, as well as specific packing business rules can be expressed, e.g.

```
gravity(Items) --> forall(O1,Items, origin(O1,3)=0
        or exists(O2,Items, uid(O1)\#uid(O2) and on\_top(O1,O2))).
```

W.-J. van Hoeve and J.N. Hooker (Eds.): CPAIOR 2009, LNCS 5547, pp. 321–322, 2009.

The search is specified here with a simple labeling of the (coordinate) variables of the items. Heuristics can be defined as preference orderings criteria, using the expressive power of the language by pattern matching. For instance, the statement `variable_ordering([greatest(volume(^ )), is(z(^ ))])` expresses the choice of the variables belonging to the objects of greatest volume first, and among them, the z coordinate first.

On Korf's benchmarks of optimal rectangle packing problems [2] (i.e. finding the smallest rectangle containing n squares of sizes $S_i = i$ for $1 \leq i \leq n$), compared to the SICStus Prolog program of Simonis and O'Sullivan [3] which improved best known runtimes up to a factor of 300, the SICStus Prolog program generated by the Rules2CP compiler explores exactly the same search space, and is slower by a factor less than 3 due to the interpretation overhead for the dynamic search predicates. The following table shows the compilation and running times in seconds:

N	R2CP compilation	Rules2CP	Original
18	0.266	13	6
20	0.320	20	10
22	0.369	364	197
24	0.443	5230	1847
25	0.509	52909	17807

Rules2CP differs from OPL, Zinc and Essence modelling languages in several aspects among which: the use of logical rules, the absence of recursion, the restriction to simple data structures of records and enumerated lists, the specification of search by logical formulae, the specification of heuristics as preference orderings, and the absence of program annotations. The expression of complex search strategies and heuristics is currently not expressible in Zinc and Essence, and can be achieved in OPL in a less declarative manner by programming. On the other hand, we have not considered the compilation of Rules2CP to other solvers such as local search, or mixed integer linear programs, as has been done for OPL and Zinc systems.

As for future work, a large subset of PKML rules has been shown in [4] to be compilable with indexicals *within* the geometrical kernel of the global constraint `geost` providing better performance than by reification. The generality of this approach will be explored with greater generality for Rules2CP. The specification of search strategies in Rules2CP will also be explored more systematically, possibly with adaptive strategies in which the dynamic criteria depend on execution profiling criteria.

References

1. Fages, F., Martin, J.: From rules to constraint programs with the Rules2CP modeling language. In: Proc. 13th International Workshop on Constraint Solving and Constraint Programming CSCLP 2008. Recent Advances in Constraints, Roma, Italy. LNCS (LNAI). Springer, Heidelberg (to appear) (2008)
2. Korf, R.E.: Optimal rectangle packing: New results. In: ICAPS, pp. 142–149 (2004)
3. Simonis, H., O'Sullivan, B.: Using global constraints for rectangle packing. In: Proceedings of the first Workshop on Bin Packing and Placement Constraints BPPC 2008, associated to CPAIOR 2008 (2008)
4. Carlsson, M., Beldiceanu, N., Martin, J.: A geometric constraint over k-dimensional objects and shapes subject to business rules. In: Stuckey, P.J. (ed.) CP 2008. LNCS, vol. 5202, pp. 220–234. Springer, Heidelberg (2008)