Documention of the scheduling zoo

The zoo keepers

December 16, 2015

1 Vocabulary

A problem is specified in the $\alpha|\beta|\gamma$ notation. Since its introduction, researchers extended it to include more and more problems. Today the notation is not standardized. In the literature a problem can be found denoted with different notations. In particular the two books from the community, by Pinedo, and by Brucker and Knust use different notations. We try to meet both notations in order to include in the zoo results from these authors.

In an abstract language, the notation consists of fields, and each field can have different values. The presence of some fields can be conditioned on the presence on some values. For example the field `processing time` can have the value $p_j = p$ only when the value P is present in the problem specification, and does not make sense in the Q execution environment for example, where the value $p_{ij} = p$ is to be used.

2 Syntax of the notation specification

The notation is specified in an XML document, called `notation.xml`. This document is both used to generate the HTML form of the zoo and the search engine. The generator is a python program called `extract.py`, which can take different arguments to choose the generated output. A `Makefile` generates all the necessary files.

The root of the XML document is called `notation` and has exactly 3 childs: `aliases`, `form` and `reductions`.

The `aliases` node contains tags with text constants to be replaced in the explanation and the requires attributes.

Each child of the `form` node is a `section` node, which corresponds basically to one of the α, β, γ and is translated into an H2 HTML title.

Optional section attribute `add_separator` is used to insert the pipe ”—” in the problem name.

A section contains several fields. Each field has a name as it appears in the HTML file. Field attribute `mandatory="True"` specifies that a value needs to be selected for that field. Otherwise leaving it blank is a valid option. The attribute `hide` is used for the interface selection field, which should not be part of the problem name. Another attribute `separation="False"` can be added, to specify that this field should not be separated from the previous field by a semicolon in the problem name. This applies to the number of machines as in `P3` and to the time lags as in `prec(l)`.

A field contains several choices that translate into a pop choice for that field in the HTML form. Every choice has a name. This name should be written in LaTeX formatting, as in an inline math formula. The $<$ sign is problematic in an XML file, also we decided to use only $\leq$ and $>$ as comparators.A choice name can contain spaces, but should not have two consecutive spaces, nor trailing or leading spaces.

A choice node should have an explanation attribute, which contains HTML code to be shown in the yellow explanation box. This code can have math formulas between $ signs, which will be rendered using MathJax.

Both field and choice nodes can have a `requires` attribute. This specifies under which condition this choice or field can be selected. The condition is a boolean expression using keywords `not`, `and`, `or` and parenthesis which have to be surrounded by spaces for simple parsing (and to distinguish them from parenthesis inside values, as in `prec(l)`).

A word in this expression (which is not a keyword) is evaluated to True either if it is present in the problem name, meaning it has been selected in some field in the HTML form, or if it is a field name with a non empty selected choice. If the expression is false, the form or the choice is hidden from the webpage. If all choices of a field are hidden, the field itself is hidden as well. It is ok for words in this expression to have spaces as in `advanced and precedence relation`.

The `reductions` node consists of a sequence of `reduction` nodes, each with an attribute `from` and `to`. The given values must belong to the same field as specified in the `form` part above. It means that a problem with the from value is a particular case of the problem with the to value. An optional attribute `requires` specifies a condition under which this reduction is valid. Transitive reductions do not to be specified explicitly. An attribute `explanation` can be used to document this reduction.

More sophisticated reduction rules could be possible but we restricted to those for the ease of the implementation.

3 Syntax of the bibtex files

The bibtex files should be written with UTF8 encoding, but maybe it is better if accents are avoided. Long bibtex identifiers are preferred in the form Authors:year:title-prefix. The BibDesk program can automatically generate those identifiers.

Results are mentioned in the Annote field, separated by newlines. Trailing double backslashes will be removed if present. A result is expected to be of the form `$problem-name$ result`, where `result` is a text in an unrestricted format. Whenever the text contains one of the strings `NP,` `hard,` `\geq` or ¿= then it is considered as a negative result. Otherwise it is considered as a positive result.

A negative result could be an NP-hardness proof, or APX-hardness proof, or a lower bound on the approximation ratio or on the competitive ratio. A positive result could be some algorithm, one could even mention available benchmarks or implementations here.

4 Reduction of problems

When we want to find out whether problem A is a particular case of problem B, then first the problems are translated into dictionaries, mapping every field name to the corresponding value in the problem name. Then A is a particular case of B if for each field, the corresponding value in A reduces to the corresponding value in B. The optimal condition given by the reduction applies to values present in A.

5 Indices

We choose j for jobs, and i for machines, and k for the operations of a job in the flow job problem.

6 Multipurpose machines

There is this model where a job j can only be scheduled on one out of an allowed set of machines $S_j \subseteq \{1, \ldots, m\}$. Such a model is a special case of the unrelated machine environment R with the restriction $p_{ij} \in \{p_j, \infty\}$. We decide for the later notation. BTW, this model is often used to show lower bounds for the unrelated machine model, and even though it is a special case often it captures the whole difficulty of the model.

7 Preemption

Preemption is a natural in the presence of release times. However there are a bunch of results without release times and preemption, in the context of flow job or precedence relation on identical parallel machines. We use the notation `pmtn`.

8 To be added later

- Job families $fmls$ from Pinedo p.32.
- Speed-scaling.
- Particular objective values, such as earliness plus tardiness around a common due date.