PERA package worksheet (1.0)

Jérôme Euzenat
INRIA Rhône-Alpes
Jerome.Euzenat@inrialpes.fr

This worksheet displays the capabilities of the PERA Maple package. A package dedicated to the algebras of relations extended with a neighborhood structure.
In particular it generates the algbras presented in the [Euzenat 1998a] paper.

First load the program of PERA:

```
> read(`Ar sorserez:Programmes:Granularitemps:Maple programs:PERA:pera`):
```

Playing with instant algebras

Then load the definition of basic instant algebre (for the sake of examples, they could be given by hand):

```
> read(`Ar sorserez:Programmes:Granularitemps:Maple programs:PERA:a3xx`):
```

The curalg variable corresponds to the algebra taken as current for the simplification algorithm. It is thus important and can lead to incorrect results if the current algebra is not the proper one. Let first consider the instant algebra (3).

```
> curalg := 3:
```

The instant algebra can be inspected by printing its base, set of auto inverse relations, composition table (which is better displayed as a matrix) and neighborhood relation (would be better displayed as a graph).

```
> print(algebra[3][base]);
```

$$\{ equals, anterior \}$$

```
> print(algebra[3][autoinverse]);
```

$$\{ equals \}$$

```
> printcompasarray(3);
```

$$[3, equals, anterior, anterior^i]$$
$$[equals, \{equals\}, \{anterior\}, \{anterior^i\}]$$
$$[anterior, \{anterior\}, \{anterior\},$$
$$\{equals, anterior, anterior^i\}]$$
$$[anterior^i, \{anterior^i\},$$
$$\{equals, anterior, anterior^i\}, \{anterior^i\}]$$
$$3$$

```
> print(algebra[3][neighborhood]);
```

$$\{[anterior, equals], [equals, anterior^i]\}$$

It is also possible to actually use the algebra for computing inverse:

```
> i(anterior);
```

$$anterior^i$$

```
> i(i(i(anterior)));
```

$$anterior^i$$

```
> i(equals);
```

$$equals$$

composition:

```
> compose(3,anterior,anterior);
```

$$\{anterior\}$$

```
> compose(3,i(anterior),equals);
```

$$\{anterior^i\}$$

```
> i(compose(3,anterior,anterior) union compose(3,i(anterior),equals));
```

$$\{anterior, anterior^i\}$$

```
> compose(3,i(anterior),anterior);
```

$$\{equals, anterior, anterior^i\}$$

or neighborhood (it is noteworthy that a relation is not a neighbor of itself here):

```
> neighbors(3,i(anterior));
```

$$\{ \textit{equals} \}$$

```
> neighbors(3,equals);
```

$$\{ \textit{anterior}^{\,i}, \textit{anterior} \}$$

```
> compose(3,i(neighbors(3,equals)),neighbors(3,anterior));
```

$$\{ \textit{anterior}^{\,i}, \textit{anterior} \}$$

```
> neighborp(3,equals,equals);
```

$$\text{false}$$

```
> neighborp(3,equals,anterior);
```

$$\text{true}$$

Checking that each cell of the composition table a neighborhood can be useful:

```
> checkneighborhood(3);
```

$$\text{true}$$

Several algebras have been loaded with the a3xx file. A3fp correspnd to time with a branching future and path (or partially ordered time). It is possible to restrict A3fp to a particular set of relationships. Is the restriction of A3fp truly A3 ? (here 0 means that they are equal, see source code for more explainations).

```
> curalg := fp3:
```

```
> algebra[r3] := restrictalgebra(fp3,{anterior,equals}):
```

```
> algebraequalp(algebra[r3],algebra[3]);
```

$$0$$

Is the restriction of A3f truly A3 ?

```
> algebra[rf3] := restrictalgebra(f3,{anterior,equals}):
```

```
> algebraequalp(algebra[rf3],algebra[3]);
```

$$0$$

Is the restriction of A3p truly A3 ?

```
> algebra[rp3] := restrictalgebra(p3,{anterior,equals}):
```

```
> algebraequalp(algebra[rp3],algebra[3]);
```

$$0$$

Infering interval algebra

Load the definition of genuine Allen algebra [Allen 1983] (for comparison and translation of the obtained relation purpose):

```
> read(`Ar sorserez:Programmes:Granularitemps:Maple programs:PERA:a13`):
```

Infer the interval algebra corresponding to the instant algebra (3) with anterior as the relation governing the intervals:

```
> inferintervalalgebra(3,n13,anterior):
```

The construction of Allen algebra is based on the work by Robin Hirsh [Hirsh 1996].
It can be compared with the Allen algebra. For that purpuse the interval relations obtained are renamed according to a table provided in the a13 file.

```
> curalg := 3:
```

```
> algebra[13] := renamealgebra(algebra[n13],intervaltablefp13):
```

```
> print(algebra[13][base]);
```

$$\{e, m, s, b, o, d^i, f^i\}$$

It is possible to display the neighborhood graph first presented in [Nökel 1988] (unfortunately not in a graphic format).

```
> print(algebra[13][neighborhood]);
```

$$\{[b, m], [m, o], [o, s], [s, e], [s, d], [d, f], [e, f], [o, f^i], [f^i, e], [f^i, d^i], [d^i, s^i], [s^i, o^i], [f, o^i], [e, s^i], [m^i, b^i], [o^i, m^i]\}$$

The composition table would be too long to display (even as a matrix). Thus the algebras are just compared. First, the curent algebra can be set to the new 13 (otherwise, the comparison will not recognise the auto inverse operators).

```
> curalg := 13: algebraequalp(algebra[13],algebra[genuine13]);
```

$$0$$

Compute the set of pointisable relations and their number [Van Beek 1990]:

```
> curalg := 3: nops(pointisable(n13));
```

$$73$$

Partially ordered time

This is the subject of [Euzenat 1998].
We first deal with the algebra of partially ordered time (meaning that it can branch in the future and in the past). It starts with the partially ordered instant algebra (fp3):

```
> curalg := fp3:
```

```
> inferintervalalgebra(fp3,nfp13,anterior):
```

Again, one can rename the relations according to the conventions in [Euzenat 1998].

```
> algebra[fp13] := renamealgebra(algebra[nfp13],intervaltablefp13):
```

It is possible then to inspect the obtained algebra:

```
> print(algebra[fp13][base]);
```

$$\{x, y, k, e, m, s, b, p, o, q, g, ii, h, d^i, f^i, z^i, j^i\}$$

```
> print(algebra[fp13][autoinverse]);
```

$$\{x, y, e, g, ii\}$$

```
> print(algebra[fp13][neighborhood]);
```

$$
\begin{aligned}
\{\, & [b, m], [m, o], [o, s], [s, e], [s, d], [d, f], [e, f], \\
& [o, h], [b, h], [z, p], [h, y], [b, k], [q, d], [z, ii], \\
& [z, x], [h, x], [m, p], [p, y], [q, z], [g, f], [p, z], \\
& [h, o], [p, x], [o, q], [j, q], [m, q], [j, z], [z, q], \\
& [j, g], [k, s], [k, q], [j, d], [h, ii], [k, h], [j, ii], \\
& [j, x], [k, p], [k, j], [j, o], [x, q^i], [z^i, h^i], \\
& [p^i, h^i], [o^i, j^i], [ii, h^i], [h^i, k^i], [ii, j^i], [g, j^i], \\
& [p^i, k^i], [q^i, k^i], [k, f^i], [f, k^i], [j^i, k^i], [d^i, j^i], \\
& [y, s^i], [x, p^i], [j^i, o^i], [g, q^i], [p^i, o^i], [f^i, g], \\
& [q, z^i], [z^i, p^i], [z, q^i], [o^i, k^i], [s, y], [p^i, m^i], \\
& [ii, z^i], [x, z^i], [h, d^i], [y, h^i], [q^i, m^i], [d, h^i], \\
& [x, h^i], [z^i, q^i], [o^i, h^i], [y, p^i], [j^i, b^i], [q^i, j^i], \\
& [d, p^i], [p, d^i], [q^i, o^i], [z^i, j^i], [x, j^i], [s^i, k^i], \\
& [o, f^i], [f^i, e], [f^i, d^i], [d^i, s^i], [s^i, o^i], [f, o^i], \\
& [e, s^i], [m^i, b^i], [o^i, m^i], [h^i, b^i], [p, z^i], [k^i, b^i], \\
& [z, p^i], [p^i, z^i], [h^i, o^i], [q^i, z^i], [d^i, q^i], [o, p], \\
& [o, j], [q, g], [q, x], [b, j], [k, o], [h, z], [h, p] \,\}
\end{aligned}
$$

It is also possible to answer the traditionnal questions:
Does A13fp restrict to A13?

```
> curalg := fp13:
```

```
> algebra[rfp13] := restrictalgebra(fp13,{b,m,o,i(d),s,i(f),e}):
```

```
> curalg := genuine13:
```

```
> algebraequalp(algebra[rfp13],algebra[genuine13]);
```

$$0$$

The answer is yes.

It is also possible to establish the number of pointisable relations available in A13FP.

```
> curalg := fp3: nops(pointisable(nfp13));
```

$$1054$$

```
>
```

Branching time

This is the subject of [Euzenat 1998].
Here time can branch in the future or in the past only. The generating principles are the same as those of above. It starts with the branching instant algebras (p3 and f3). This is presented here only with past branching time:

```
> curalg := p3:
> inferintervalalgebra(p3,np13,anterior):
> algebra[p13] := renamealgebra(algebra[np13],intervaltablefp13):
```

It is possible then to inspect the obtained algebras:

```
> print(algebra[p13][base]);
```

$$\{e, m, s, b, o, q, g, ii, d^i, f^i, j^i\}$$

```
> print(algebra[p13][autoinverse]);
```

$$\{e, g, ii\}$$

```
> print(algebra[p13][neighborhood]);
```

$$\{[g, q^i], [f^i, g], [o^i, j^i], [ii, j^i], [g, j^i], [d^i, j^i], [j^i, o^i], [d^i, q^i], [q^i, m^i], [j^i, b^i], [q^i, j^i], [q^i, o^i], [d^i, s^i], [s^i, o^i], [f, o^i], [e, s^i], [m^i, b^i], [o^i, m^i], [o, f^i], [f^i, e], [f^i, d^i], [b, m], [m, o], [o, s], [s, e], [s, d], [d, f], [e, f], [q, d], [g, f], [o, q], [j, q], [m, q], [j, g], [j, d], [j, ii], [j, o], [o, j], [q, g], [b, j]\}$$

It is also possible to answer the traditionnal questions:
Does A13fp restrict to A13?

```
> curalg := p13:
> algebra[rp13] := restrictalgebra(p13,{b,m,o,i(d),s,i(f),e}):
> curalg := genuine13:
> algebraequalp(algebra[rp13],algebra[genuine13]);
```

$$0$$

The answer is yes. But it is not the case that p13 is a restriction of fp13:

```
> curalg := fp13:
> algebra[rfp13p] := restrictalgebra(fp13,{b,m,o,i(d),s,i(f),e,g,i(j),q}):
> curalg := p13:
> algebraequalp(algebra[rfp13p],algebra[p13]);
```

$$1$$

```
>
```

References

[Allen 1983] J. Allen, Maintaining knowledge about temporal intervals, Communications of the ACM, Vol. 26, pp. 832-843, 1983.

[Euzenat 1995] J. Euzenat, An algebraic approach to granularity in qualitative time and space representation, Actes 14th IJCAI, Montréal (CA), pp. 894-900, 1995.

[Euzenat 1998] J. Euzenat, Algèbres d'intervalles sur des domaines temporels arborescents, Actes Xth RFIA, Clermont-Ferrand (FR), pp. -, 1998

[Freksa 1992] C. Freksa, Temporal reasoning based on semi-intervals, Artificial intelligence, Vol. 54, pp. 199-227, 1992.

[Hirsch 1996] R. Hirsch, Relation algebras of intervals, Artificial intelligence, Vol. 83, pp. 267-295, 1996.

[Ligozat 1994] G. Ligozat, Towards a general characterization of conceptual neighborhoods in temporal and spatial reasoning, dans Frank Anger, Rasiah Loganantharaj (éds.), actes 12th AAAI workshop on spatial and temporal reasoning, Seattle (WA US), pp55-59, 1994.

[Nökel 1988] K. Nökel, Convex relations between time intervals, Rapport de recherche 17, SEKI, Kaiserslautern (DE), 1988

[Randell, Cohn, Cui 1992] D. Randell, A. Cohn, Z. Cui, Computing transitivity tables: a challenge for automated theorem provers, Lecture notes in computer science, Vol. 607, pp. 786-790, 1992.