On computing kernels on fuzzy simple graphs by combinatorial enumeration using a CLP(FD) system

Raymond BISDORFF
CRP-University Centre, Luxembourg

1996

Abstract

This paper reports our communication done at the 8th Benelux Workshop on Logic Programming in Louvain-la-Neuve, 9 September 1996. We will present a constraint formulation in finite domains of the kernel construction on simple graphs and give some comments on implementation in CHIP. Application to fuzzy choice procedures will illustrate the theoretical developments.

1 Introducing crisp kernels

Let $A = \{a_0, a_1, \ldots, a_n\}$ be a finite set of dimension n and $R \subseteq A \times A$ a simple crisp relation on A. We call the couple $G = (A, R)$ a simple graph. A kernel of a simple graph $G = (A, R)$ is defined as a conjointly interior and exterior stable subset of A.

A given homogeneous relation R may be represented as a boolean matrix of dimension $n \times n$. The opposite or inverse relation R^{-1} is given by the transposed matrix R^t and the negated relation $\neg R$ is denoted as $\overline{R}$, that is the 1-complemented matrix. Composition of conformable relations is represented by the boolean product $\circ$ of their associated matrices and finally, crisp implication is represented by the associated subset ordering '$\leq$' on the boolean matrices.

Let $G = (A, R)$ be a simple graph and Y_R a subset of A:

$$\begin{array}{llrcl}
Y_R \text{ is interior right stable} & \Leftrightarrow & R' \circ Y_R & \leq & \overline{Y_R}, \\
Y_R \text{ is interior left stable} & \Leftrightarrow & R'^t \circ Y_R & \leq & \overline{Y_R}, \\
Y_R \text{ is exterior absorbent) (right) stable} & \Leftrightarrow & R' \circ Y_R & \geq & \overline{Y_R}, \\
Y_R \text{ is exterior dominant (left) stable} & \Leftrightarrow & R'^t \circ Y_R & \geq & \overline{Y_R}.
\end{array}$$

where matrix R' is constructed from R in the following way: $\forall a, b \in A$ with $(a \neq b), R'(a, b) = R(a, b)$ and $\forall a \in A, R'(a, a) = 0$, where R'^t is the transposed matrix R' and where $\overline{Y_R}$ is the 1-complement of Y_R.

We may now propose four different conjunctions of interior and exterior stability conditions as required by the kernel concept. Let $G = (A, R)$ be a simple graph and K_R respectively Y_R a subset of A:

$$\begin{aligned}
&K_R \text{ is a right absorbent kernel } \equiv K_R \in \Upsilon_R{}^{ar} = \{Y_R : R' \circ Y_R = \overline{Y_R}\},\\
&K_R \text{ is a right dominant kernel } \equiv\\
&\qquad K_R \in \Upsilon_R{}^{dr} = \{Y_R : (R' \circ Y_R \leq \overline{Y_R}) \wedge (R'^t \circ Y_R \geq \overline{Y_R})\},\\
&K_R \text{ is a left absorbent kernel } \equiv\\
&\qquad K_R \in \Upsilon_R{}^{al} = \{Y_R : (R'^t \circ Y_R \leq \overline{Y_R}) \wedge (R' \circ Y_R \geq \overline{Y_R})\},\\
&K_R \text{ is a left dominant kernel } \equiv K_R \in \Upsilon_R{}^{dl} = \{Y_R : R'^t \circ Y_R = \overline{Y_R}\}.
\end{aligned}$$

Taking the set union of admissible solutions for each of these relational constraints systems gives the dominant, respective absorbent kernels.

$$\begin{aligned}
&K_R \text{ is a dominant (initial) kernel } &\Leftrightarrow\quad K_R \in \Upsilon_R{}^{d} = \{\Upsilon_R{}^{dr} \cup \Upsilon_R{}^{dl}\},\\
&K_R \text{ is an absorbent (terminal) kernel } &\Leftrightarrow\quad K_R \in \Upsilon_R{}^{a} = \{\Upsilon_R{}^{ar} \cup \Upsilon_R{}^{al}\}.
\end{aligned}$$

Terminal kernels on simple graphs were originally introduced by J. Von Neumann and O. Morgenstern (1944) under the name 'game solution' in the context of game theory. J. Riguet(1948) introduced the name 'noyau (kernel)' for the Von Neumann 'game solution' and B. Roy (1958, 1969) introduced the reversed terminal or initial kernel construction as possible dominant choice procedure in the context of the multi-criteria Electre decision methods. Terminal kernels were studied by C. Berge (1958, 1970) in the context of the Nim game modelling. Recent results on such terminal kernels on graphs for solutions of different games have been reported by G. Schmidt and T. Ströhlein (1985). Fuzzy initial or dominant kernels were very recently deeply investigated by L. Kitaïnik (1993).

In this paper we will extend this crisp kernel concept to fuzzy simple graphs.

2 Generalizing kernels to fuzzy simple graphs

In order to illustrate this extension, we will first shortly introduce an algebraic framework for a fuzzy truth calculus applied to our relational calculus. In a second step we will extend the crisp kernel constructive definition to this fuzzy case. In a third part, we will shortly discuss the computational complexity of the fuzzy kernel construction.

2.1 Basic semantics for a fuzzy truth calculus

To be able to manipulate the truth of a particular relation between elements of some finite sets, we choose to introduce a completely ordered discrete set V of degrees of truthfulness conveniently represented by the rational unit interval, that is the bounded unit segment of the rational line: $V = \{0, \frac{1}{m}, \frac{2}{m}, \ldots, \frac{m}{2m} = \frac{1}{2}, \ldots, \frac{m-1}{m}, \frac{m}{m} = 1\}$, where m is any finite positive integer (generally $m = 100$, so that truth values may be seen as percents). We suppose that V supports $\wedge$ (meet) and $\vee$ (join) operations, implementing a convenient fuzzy conjunction or t-norm and associated fuzzy disjunction or t-co-norm (cf. Fodor & Roubens 1994). In order to be able to deal with multi-valued logical negation, as needed in the kernel construction, we will furthermore assume on V an unary strong symmetric negation or contradiction operation '$\neg$' defined as a strict antitonic order reversing bijection on V, such that the operator triplet $(\wedge, \vee, \neg)$ gives a De Morgan triplet on V and, if present, $\frac{1}{2}$ is the unique $\neg$-fix-point in V. We shall introduce on V an implication operator defined as binary operation '$\rightarrow$' on V being antitonic in the first and isotonic in the second argument and verifying the following condition: $\forall u \in V : u \rightarrow u = 1$. Concrete candidates for such an implication operator are for instance the classic Zadeh inclusion or the Gödel-Brouwer or residual implication. We shall call the complete algebraic structure $\mathcal{L} = (V, \leq, \wedge, \vee, \neg, \rightarrow, 0, \frac{1}{2}, 1)$ a *symmetric truth evaluation domain* (s.t.e.d.) for our relational truth calculus. The size of a given s.t.e.d. is determined by the cardinal number n of set V, that is $\mid V \mid = n$ which we indicate as $\mathcal{L}_n$.

We may now introduce $\mathcal{L}$-valued binary relations ($\mathcal{L}$-vbr for short) between two finite sets A and B, as a function R from the cartesian product $A \times B$ to the set V of $\mathcal{L}$-valued truth values. And finally a fuzzy simple graph $G^{\mathcal{L}} = (A, R)$ is given by an $\mathcal{L}$-valued homogeneous relation on a given finite set A.

A concrete examples may be shown with the help of the following diagram:

$$\begin{tikzcd}
G \quad = & a \arrow[dr, bend right, "1"'] & & b \arrow[ll, dotted, bend right, ".55"'] \arrow[dl, bend left, "1"] \\
& & c \arrow[ul, dotted, ".55"] \arrow[ur, dotted, ".55"'] &
\end{tikzcd}$$

where $A = \{a, b, c\}$ and the relation R is evaluated in a commonly used s.t.e.d. $\mathcal{L}_{100}$ with fuzzy truth values expressed as percents of credibility. The absence of a relation between two nodes, that is a link evaluated as 0 is generally not depicted in the diagram, that is $R(a, b) = 0$ in this example.

2.2 Defining $\mathcal{L}$-valued kernels

Let $G^{\mathcal{L}} = (A, R)$ be a given fuzzy graph and $\{\kappa_R\}$ a singleton set. We assume Y_R to be an $\mathcal{L}$-valued binary relation defined on $A \times \{\kappa_R\}$, that is a function $Y_R : A \times \{\kappa_R\} \rightarrow V$, where each $Y_R(a, \kappa_R)$, $\forall a \in A$, is supposed to indicate the degree of truth associated with the proposition that the 'element a is included in the kernel κ_R'. As κ_R is a constant, we will simplify our notation by dropping the second argument and in the sequel $Y_R(a), \forall a \in A$, is to be seen as an $\mathcal{L}$-valued characteristic vector for the kernel membership function defined on a given $\mathcal{L}$-vbr R. In order to construct such an $\mathcal{L}$-valued membership relation Y_R, we define the following credibility degrees on necessary implicative stability conditions similar to those imposed for kernel constructions on crisp relations.

$$\begin{array}{lcrcl}
Y_R \text{ is right interior stable} & \equiv & R' \circ Y_R & \leq & \overline{Y_R}, \\
Y_R \text{ is left interior stable} & \equiv & R^{t'} \circ Y_R & \leq & \overline{Y_R}, \\
Y_R \text{ is right exterior stable} & \equiv & R' \circ Y_R & \geq & \overline{Y_R}, \\
Y_R \text{ is left exterior stable} & \equiv & R^{t'} \circ Y_R & \geq & \overline{Y_R}.
\end{array}$$

where R' is constructed from R in the following way: $\forall a, b \in A$ with $(a \neq b), R'(a, b) = R(a, b)$ and $\forall a \in A, R'(a, a) \in [0, \frac{1}{2}]$. The classic boolean matrix product is naturally extended to our $\mathcal{L}$-algebra.

In accordance with the crisp case, we may now define $\mathcal{L}$-valued kernel solutions. Let K_R and Y_R be subsets of A:

$$K_R \text{ is a right absorbent kernel } \equiv\ K_R \in {\Upsilon_R}^{ar} = \max_{(\succeq)}\{Y_R : R' \circ Y_R = \overline{Y_R}\},$$

$$K_R \text{ is a right dominant kernel } \equiv$$

$$K_R \in {\Upsilon_R}^{dr} = \max_{(\succeq)}\{Y_R : (R' \circ Y_R \leq \overline{Y_R}) \wedge ({R'}^t \circ Y_R \geq \overline{Y_R})\},$$

$$K_R \text{ is a left absorbent kernel } \equiv$$

$$K_R \in {\Upsilon_R}^{al} = \max_{(\succeq)}\{Y_R : ({R'}^t \circ Y_R \leq \overline{Y_R}) \wedge (R' \circ Y_R \geq \overline{Y_R})\},$$

$$K_R \text{ is a left dominant kernel } \equiv\ K_R \in {\Upsilon_R}^{dl} = \max_{(\succeq)}\{Y_R : {R'}^t \circ Y_R = \overline{Y_R}\}.$$

In this construction, we say that Y is *sharper* than Y', noted $Y \succeq Y'$ iff $\forall a \in A$ either $(Y(a) \leq Y'(a) \leq \frac{1}{2})$ or $(\frac{1}{2} \leq Y(a) \leq Y'(a))$. Naturally all crisp kernels are maximal sharp, so that this requirement does not make sense in the crisp case. But in the general $\mathcal{L}$-valued case, we retain thus as efficient kernel solution only the sharpest ones.

Finally, we denote ${K_R}^k$ with $k = \{ra, rd, la, ld\}$ the different solution sets for the corresponding $\mathcal{L}$-valued relational inequality systems, where R' represents the $\mathcal{L}$-anti-reflexive transform of R and for a given $\mathcal{L}$-vbr R, we shall call the set ${K_R}^d = \{Y_R : Y_R = \max_{\succeq}({K_R}^{rd} \cup {K_R}^{ld}\}$ its dominant kernels and the set ${K_R}^a = \{Y_R : Y_R = \max_{\succeq}({K_R}^{ra} \cup {K_R}^{la}\}$ its absorbent kernels.

The above shown fuzzy simple graph admits two such fuzzy dominating kernel solutions: $K_1^d = [.45, 1, 0]$, $K_2^d = [.45, .45, .55]$, and two such fuzzy absorbent kernels solutions: $K_1^a = [.55, 0.45, 0.45]$, $K_2^a = [0, 0, 1]$.

Before presenting a Prolog implementation of our kernel calculus, we will shortly discuss the computational complexity of our formal constructions.

2.3 Complexity of the $\mathcal{L}$-valued kernel construction

Let R be a $\mathcal{L}$-vbr defined on finite set $A = a_1, a_2, \ldots, a_i, \ldots, a_n$ with $\mathcal{L} = (V, \leq, \wedge, \vee, \neg, \rightarrow, 0, \frac{1}{2}, 1)$ being a s.t.e.d. of size $2m+1$, that is the set V is of finite dimension $2m+1$. Computing the sets K_R of maximal kernel $\mathcal{L}$-valued characteristic vectors defined on R will necessitate to solve several constraint satisfaction problems (CSP's) of size $\mathcal{O}((2m+1)^n)$. The CSP class of decision problems is known to be NP-complete and in general for even small finite s.t.e.d.'s, enumeration procedures to compute the sets K_R will not be practically achievable.

In the crisp case however, the corresponding CSP's are of size $\mathcal{O}(2^n)$ and a clever enumeration procedures may give results for small finite $\mathcal{L}$-vbr's. But for acyclic relations, efficient constructive algorithms to exhibit all dominant or absorbent kernels for a given crisp relation R do exist (Roy 1969). For complete $\mathcal{L}$-vbr's, there exists an algebraic computation method of linear complexity $\mathcal{O}(n)$ for lower bounds in fuzziness of the n corresponding maximal $\mathcal{L}$-valued kernels. The stability of this method w.r.t. the strong $\mathcal{L}$-negation allows to use a similar approach of linear complexity $\mathcal{O}(n)$ to compute lower bounds of fuzziness for the left and right unique kernel solutions in the case of $\mathcal{L}$-empty relations (Bisdorff &Roubens 1996a).

In the presence of general $\mathcal{L}$-valued relations with associated infinite s.t.e.d. $\mathcal{L}$, as given for instance by the rational [0, 1] interval, there does not exist any known direct algebraic procedure for the kernel construction, except for the one-dimensional case. For the class of mononuclear crisp relations, that is with a unique maximal kernel solution, there exists a fix-point method, initially invented by Von Neumann, to compute the unique dominant or absorbent $\mathcal{L}$-valued kernel in a convergence depending linearly on the size of the relation underlying set A. A variant of this method has been adapted or invented by B. Roy for the kernel construction in his Electre methods (Roy & Bouyssou 1993). The fix-point approach is extensively discussed by Schmidt & Ströhlein (1989) and has been reinvented or adapted by Kitaïnik to the fuzzy case (1993).

Practical experimentation however has shown that maximal kernel solutions for general $\mathcal{L}$-vbr's are always computable in a restricted finite minimal $\mathcal{L}_{FD}$ sub-algebra of $\mathcal{L}$ generated by the finite subset of degrees of credibility underlying the evaluation of a given finite $\mathcal{L}$-vbr R (Bisdorff & Roubens 1996).

But instead of determining all dominant or absorbent kernel solutions, it is generally more efficient to search immediately for the most discriminating kernel solution in the sense that it will thus have the overall highest degrees of truth for kernel-membership and by the same the overall lowest degrees of truth for not kernel-membership. In other words, we directly look for the overall sharpest maximal kernels.

Formally, assume R to be an $\mathcal{L}$-vbr defined on a finite set A and taking values in a given s.t.e.d. and let $\mathcal{L}_R$ be the associated restricted minimal truth evaluation domain. Let ${K_R}^k$ be the sets of all admissible $\mathcal{L}_R$-valued kernel characteristic vectors where $k = \{ra, rd, la, ld\}$. We define on $\forall Y_R \in {K_R}^k$ a sharpness index σ^2 as follows:

$$\sigma^2 = \sum_{a \in A} (Y(a) - \frac{1}{2})^2$$

Thus, choosing a most sharp or optimal $\widehat{K_R}$ $\mathcal{L}$-valued kernel solution for any of the four partial kernel construction, may be summarised as follows:

$$\widehat{K_R} = \max_{Y_R \in {K_R}^k} (\sigma^2(Y_R))$$

And determining the $\mathcal{L}$-valued *optimal* kernel solutions, appears to be a combinatorial constraint optimisation problem.

3 CLP(FD) implementation of the kernel construction

In order to be able to solve in reasonable time and space limits such enumeration problems, we propose to use a finite domain solver of type CLP(FD), i.e. a finite domain solver embedded in a constraint logic programming system. As generally noticed (Jaffar 1994), such finite domains solvers appear to be particularly useful as a formal specification and implementation tools for combinatorial optimisation problems defined on finite domains (Dincbas & al. 1990). We choose as specific CLP system, the CHIP (Constraint Handling In Prolog) system (Aggoun & al. 1991). Our CHIP $\mathcal{L}_{FD}$ model will be presented briefly below.

3.1 CHIP formulation

The set A, the evaluation domain underlying integer number set V and the $\mathcal{L}_{FD}$-vbr R is formulated in Prolog as follows:

```
        action_set([ a1, a2, a3, a4, a5, a6, a7, a8]).
        evaluation_domain(V) :- L :: 0..100, dom(L, V).
        relation(a1, a1, 0).
        relation(a1, a3, 70).
        relation(a1, a4, 62).
         ...
```

The $\mathcal{L}_{FD}$ lattice operators are implemented by two primitive predicates `minimum` and `maximum` and the corresponding $\mathcal{L}$-negation is implemented as additive complementing. The crisp true part of the $\mathcal{L}$-implication corresponds to the natural relational inequality operators on finite domains.

The predicate `kernel_solution` calculates kernel solutions $Y_R \in {K_R}^k$ for $k = \{ra, rd, la, ld\}$ on the basis of a valid data set. We will illustrate the program on the right dominant case. The other cases are similar and the fully implemented Prolog programs are actually parametric to produce the four possible kernel solutions.

```
kernel_solution(Relation, Y):-
  [-Relation],
    % include the data set predicates
  action_set(A),
    % get the elements of set A
  length(A, N)
  evaluation_domain(V),
  Min :: V, minimum(Min, V),
  Max :: V, maximum(Max, V),
    % get the evaluation domain underlying truth values
    % with Min and Max the bottom, respective the top element
  length(Y, N), Y :: V,
    % constructing the kernel solution vector with
    % appropriate value domains

  construct_relation(N, A, A, 0, [], R),
    % constructing the  L_FD-anti-reflexive representation R'
  prodmat(R, Y, [], X1),
    % composing the relation R' with Y to give X1
  length(Yn, N), Yn :: V,
  negation(Min, Max, Y, Yn),
    % constructing the L_FD-negation Yn of Y
  inequality(X1, Yn),
    % right interior stability constraint

  construct_relation(N, A, A, 1, [], Rt),
    % constructing the reversed (transposed) relation
  prodmat(Rt, Y, [], X2),
    % composing the transposed relation Rt with Y
```

```
inequality(Yn, X2),
  % dominant stability constraint

labelling(Y, 0, first_fail, indomain(Y, max)).
  % conditional enumeration with a first_fail instantiating
  % order and starting with highest possible truth values
  % first.
```

The declarative aspect of Prolog programming style follows closely the mathematical formulation of the problem and the code is in itself illustrating.

For small $\mathcal{L}_{FD}$-vbr's we may use the above `kernel_solution` predicate in conjunction with the Prolog primitive `findall` for collecting the set K^k of all admissible kernel solutions:

```
findall(Y, kernel_solution(Relation, Y), Kk),
```

on which we apply a special sorting algorithm to filter out all maximal sharpest solutions in the sense of relation '$\succeq$'. Re-applying the same sorting algorithm to the union of the resulting right and left dominant kernel solutions or to the union of the corresponding right and left absorbent kernel solutions gives finally the effective dominant, respective absorbent, maximal fuzzy kernels. Unfortunately, this straight forward or 'brute force' solving procedure is only operational for small relations or for small evaluation domains as in the crisp case for instance.

For larger relations, we may use the CHIP inbuilt branch and bound enumeration predicate `min_max` to search directly, with the help of our overall sharpness index σ^2, an optimal discriminating dominant or absorbent kernel solution.

3.2 Implementing the stability constraints

But, what makes our CHIP implementation generally interesting, from an operational point of view, is the embedded dynamic propagation mechanism for binary linear and interval constraints (Van Hentenryck & al. 1989). This propagation is based on techniques for node- and arc-consistency and interval propagation on the constraints graph associated with the finite domain variables `Y`, `Yn`, `X1` and `X2` (cf. Jaffar & al. 1994).

To illustrate the formal specification of this mechanism, we will present in some detail the `prodmat`, `negation` and `inequality` predicates.

```
prodmat([], _, Z, Z) :- !.
prodmat([M|Tm], Q, Z, Z3):-
       prodvectmat(M, Q, [], Z1),
       append(Z, [Z1], Z2),
       prodmat(Tm, Q, Z2, Z3).
```

If the first input matrix is empty the predicate gives back an unchanged result. Otherwise the first evaluation vector is composed with the second relation matrix. And this process is recursively repeated for all relation vectors of the first relation matrix.

The composition of an evaluation vector with a relation matrix is done by the `prodvectmat` predicate:

```
prodvectmat(_, [], Z, Z) :- !.
prodvectmat(M, [Q|Tq], Z, Z2) :-
    evaluation_domain(V),
    [X, X1] :: V, minimum(X, V),
       % initialisation of the result X to the bottom element;
    prodvect(M, Q, X, X1),
    append(Z, [X1], Z1),
    prodvectmat(M, Tq, Z1, Z2).
```

If the input vector is empty, the predicate returns an unchanged result. Otherwise, the result of vector composition is stored in the X1 variable and this calculus is recursively repeated for all vectors of the second argument. The composition of two evaluation vectors is achieved by the prodvect predicate.

```
prodvect(_, [], X, X) :- !.
prodvect([M|Tm], [Q|Tq], X, X2) :-
        maxmin(M, Q, X, X1),
        prodvect(Tm, Tq, X1, X2).

maxmin(M, Q, X, X2) :-
        evaluation_domain(V),
        [X1, X2] :: V,
        minimum(X1, [M, Q] ),
        maximum(X2, [X, X1]).
```

If the vectors on input are empty, the evaluation stays unchanged. Otherwise, the maxmin predicate computes the actual valuation by using primitive maximum and minimum predicates implementing dynamic constraints on the unbounded resulting domain variables.

The negation predicate implements the $\mathcal{L}_{FD}$-negation on a truth evaluation vector Y :

```
negation(_, _, [], []) :- !.
negation(Min, Max, [Y|Ty], [Yn|Tyn]) :-
        Y + Yn #= Max + Min,
        negation(Min, Max, Ty, Tyn).
```

If the first input relation vector is empty, the predicate gives back an empty result. Otherwise the first component of the result Yn is additively complemented with the first component of Y. And this process is recursively repeated for all the components of the input vector Y. The relational FD-operator '#=' installs an equality constraint on the corresponding domain variables. Thus, at instantiation of Y, dynamic propagation of these domain equations result in automatically implied instantiation of Yn.

The inequality predicate implements in a same manner the implicative inequalities for credibility vectors.

```
inequality([], []):- !.
inequality([X|Tx], [Y|Ty]) :-
        X #<= Y,
        inequality(Tx, Ty).
```

If the input relation vectors are empty the predicate does nothing. Otherwise the first domain variable `X` is constrained through the relational FD-operator '`#<=`' to take lower or equal values to the second domain variable `Y`. And this process is recursively repeated for all domain variables of the input vectors `X` and `Y`. Thus again, at instantiation, dynamic propagation of these domain inequations results in automatically implied limits on the values taken respectively by `X` and `Y`.

4 Practical application to fuzzy choice

To illustrate the operational performances of our CHIP implementation for general $\mathcal{L}$-valued kernel constructions, we choose a concrete example out of the context of preference modelling, where constructing choice functions on degrees of credibility associated to out-ranking relations on a set A of alternatives through an out-ranking index (cf. Roy & Bouyssou 1993) is a natural candidate for the search of kernels on non trivial $\mathcal{L}$-vbr's. Our computation are done on a Cray CS6400 Super-server under Solaris 2 with the CHIP V.4.1.0 version.

The considered set A contains 8 alternatives: $A = \{a, b, c, d, e, f, g, h\}$. The $\mathcal{L}$-vbr R defined on this set A corresponds to an out-ranking index, constructed as two digits decimal truth values for the out-ranking relation supposed to hold on the set of alternatives (Roy & Bouyssou 1993). Here R is represented as a valued matrix in a Prolog programming syntax as used in our CHIP implementation. The dummy reflexive part '_' of the relation is confined to take only $\mathcal{L}$-anti-reflexive values.

```
R = [[ _, 75, 70,  62,   0,  0,  0,  0],
     [76,  _, 90, 100,  82, 82, 82, 80],
     [70, 86,  _,  100, 100, 46, 80, 91],
     [64, 65, 94,   _,  88, 22, 94, 74],
     [33, 57, 93, 100,   _,  0, 80, 86],
     [ 0, 73, 64,  92,  76,  _, 96, 80],
     [ 0, 63, 73,  85,  82, 70,  _, 81],
     [ 0, 60, 64,  60,  77,  0,  0,  _]].
```

The associated integer evaluation domain uses truth values expressed as two digits integer percents, that is $V = \{0, 1, \ldots, 50, \ldots, 99, 100\}$, and for instance $R(a, b) = 75$ means here that the truth value of the proposition 'alternative a is out-ranking alternative b' is supposed to be 75%.

We obtain three maximal dominant kernel solutions:

```
Y_Rd   = [ [24, 76, 24,  24, 24, 24, 24, 24],
           [70, 30, 30,  30, 30, 30, 70, 30],
           [70, 30, 30,  30, 30, 70, 30, 30]].
```

The sharpest dominant kernel is selecting the action $\{b\}$ with credibility 76%, rejecting on the other hand all other actions with credibility $100-76 = 24\%$. The other two maximal dominant kernel solutions, equivalent in overall credibility, select either the couple $\{a, f\}$ or the couple $\{a, g\}$ with same credibility of 70%. Overall computation time is about 18 seconds with approximately 9 seconds for generating the right respectively the left dominant maximal solutions.

By the way, the CHIP primitive predicate min_max, implementing the combinatorial optimization of our overall sharpness index σ^2, returns following optimal right dominant kernel solution.

```
THE SOLUTION IS
labelling([24, 76, 24, 24, 24, 24, 24, 24], 0, first_fail, indomain)
ITS COST IS 14586
----------------------------------------
min_max -> proven optimality
```

The optimization step iterates in a kind of branch and bound mechanism through the set ${\Upsilon_R}^{rd}$, improving the kernel solution by augmenting the difference between $\mathcal{L}$-true and $\mathcal{L}$-untrue values, to return finally, in about 8 seconds, the above solution with proven optimality. The calculated optimal kernel selects again element $\{b\}$ as kernel member with a credibility of 76% and de-selects or rejects all the other elements with complement credibility of 24%. From an optimal unique choice point of view, the above overall sharpest dominant kernel thus proposes action {b} as best first choice with associated credibility of 76%.

5 Concluding remarks

Our CHIP implementation of the kernel construction allows us to compute in reasonable time and space limits kernel solutions for only small $\mathcal{L}$-valued graphs. But it allows to explore the solution space and discover its internal structure. Furthermore, the Prolog formulation is generic in the sense that any other fuzzy disjunction and conjunction as for instance the Lukasiewicz operators may be used instead of the standard max and min operators.

Research is actually ongoing in this direction and some efficient algorithms, mixing constraint enumeration and fix-point iteration have been successfully applied to fuzzy kernel construction.

Finally, it is worthwhile mentioning that the support of the CLP(FD) tool, allowed us, through its efficient constraint enumeration procedure, to uncover the interesting theoretical properties of the kernel solution spaces in standard min/max logical algebras (cf. Bisdorff & Roubens 1996).

References

[1] Aggoun, A., and Beldiceanu, N., Overview of the CHIP compiler system, in proc. *ICLP 91,MIT Press*, 1991

[2] Berge, Cl., Théorie des graphes et ses applications, *Dunod*, Paris, 1958

[3] Berge, Cl., Graphes and Hypergraphes, *Dunod*, Paris, 1970

[4] Bisdorff, R. and Roubens, M., On defining fuzzy kernels from L-valued simple graphs, *in proceedings of IPMU'96 (Information Processing and Management of Uncertainty in Knowledge-Based Systems)*, Granada, July, 1996a

[5] Bisdorff, R,and Roubens, M., On defining and computing fuzzy kernels from L-valued simple graphs, *in proceedings of FLINS'96 workshop on Intelligent Systems and Soft Computing for Nuclear Science and Industry*, Mol, Belgium, September 1996b

[6] Bolc, L. and Borowik, P., Many-valued Logics: Theoretical foundations, *Springer-Verlag*, Berlin, 1992

[7] Dincbas, M., Van Hentenryck, P., Simonis, H, Solving large combinatorial problems in logic programming *J. Logic Programming*, New York, vol. 8, N 1&2, January/ March, 1990

[8] Fodor, J. and Roubens, M., Fuzzy preference modelling and multi-criteria decision support, *Kluwer Academic Publishers*, 1994

[9] Jaffar, J. and Maher, M. J., Constraint logic programming: a survey *J. Logic Programming*, New York, vol. 19/20, pp. 503-581, May/July, 1994

[10] Kitaïnik, L., Fuzzy decision procedures with binary relations: towards a unified theory, *Kluwer Academic Publ.*, Boston, 1993

[11] Roy, B. and Bouyssou, D., Aide multicritère à la décision: Méthodes et cas, *Economica*, Paris, 1993, chap. 5

[12] Schmidt, G. and Ströhlein, Th., Relationen und Graphen, *Springer-Verlag*, Berlin, 1989

[13] Schmidt, G. and Ströhlein, Th., On kernels of graphs and solutions of games: A synopsis based on relations and fix-points, *SIAM, J. Algebraic Discrete Methods*, 6 (1985), pp. 54-65

[14] Von Neumann, J. and Morgenstern, O., Theory of games and economic behaviour, *Princeton Univ. Press*, Princeton, N.J., 1944

[15] Van Hentenryck, P., Constraint satisfaction in logic programming, *MIT Press*, 1989