Stable Matching Problems with Soft Constraints (Extended Abstract)

Maria Silvia Pini
University of Padova, Italy
Padova, Italy
pini@dei.unipd.it

Francesca Rossi
University of Padova
Padova, Italy
frossi@math.unipd.it

Kristen Brent Venable
Tulane University & IHMC, USA
kvenabl@tulane.edu

ABSTRACT

We consider stable matching problems where the agents express their preferences compactly via soft constraints. We study the impact of this choice on the computational complexity of finding stable matching, with particular attention to fuzzy constraints.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems*; F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity

General Terms

Theory, Algorithms

Keywords

Stable Matching, Preferences, Soft Constraints

1. INTRODUCTION

The stable matching (SM) problem is a well-known problem with many practical applications. It has to do with two sets of agents, often called men and women, that should be matched in such a way that no man and woman, who are not married to each other, both prefer each other to their current partner [5]. This property is called stability. Problems of this kind arise in many real-life situations, such as assigning junior doctors to hospitals, children to schools, students to campus housing, kidney transplant patients to donors, and so on. The most well-known and used algorithm to find a stable matching is the GS algorithm [4], that runs in polynomial time. This algorithm assumes that both men and women express their preferences over all members of the other gender. However, this can be unfeasible, since the number of men and women can be very large. In addition, eliciting the preferences may be a costly and time-consuming process. Fortunately, the sets of men and women may have a combinatorial structure, which allows for expressing preferences in a compact way by referring to features rather than entire men or women.

Our challenge is to understand how to adapt the GS algorithm to work with such preference statements over features, expressed via soft constraints, and to study the impact of this approach over the computational properties of the algorithm. The main operations performed by the GS algorithm are the following ones: men need to exploit their preferences over women to find their most preferred woman, and possibly also their next most preferred woman (several times), while women need to compare two men according to their preferences over men. We study what it means to perform such operations in a soft constraint setting. Moreover, soft constraints induce a total order over men and women, possibly with ties. The GS algorithm requires a strict total order (that is, no ties) over men and women. We consider three ways to break ties, with particular attention to fuzzy constraints.

A similar study was done using CP-nets instead of soft constraints [6]. However, CP-nets [1] have very different properties from soft constraints, both in terms of expressiveness and of computational complexity of reasoning with them.

Appears in: *Alessio Lomuscio, Paul Scerri, Ana Bazzan, and Michael Huhns (eds.), Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014), May 5-9, 2014, Paris, France.*

2. STABLE MATCHING PROBLEMS

The stable marriage problem (SM) [5] is a well-known problem of matching n men to n women to achieve a certain type of 'stability'. Given n men and n women, where each person expresses a strict total ordering over the members of the opposite sex, the problem is to match the men to the women such that no two people of the opposite sex, who are not married to each other, both prefer each other to their current partners. If there are no such pairs, called blocking pairs, every marriage is stable.

In [4] Gale and Shapley (GS) provided an $O(n^2)$ time algorithm for finding a stable marriage. The GS algorithm consists of a number of rounds in which each un-engaged man proposes to his most preferred woman to whom he has not yet proposed. Each woman receiving a proposal becomes "engaged", provisionally accepting the proposal from her most preferred man. In subsequent rounds, an already engaged woman can "trade up", becoming engaged to a more preferred man and rejecting a previous proposal, or, if she prefers him, she can stick with her current partner. Given a matching M, we will denote with $M(w)$ (resp., $M(m)$) the man (resp., woman) associated to the woman w (resp., man m) in M. Also, $pref(x)$ denotes the preference list of a man or a woman x. The GS algorithm includes the following operations: $Opt(pref(m))$: Computes the optimal woman for m (i.e., m's first proposal); $Next(pref(m), w)$: computes the next best woman after w for man m (i.e., a new proposal for m); $Compare(pref(w), m, m')$: returns true if woman w prefers man m to m'. This is needed when woman w, currently matched with m', must decide whether to accept or decline a proposal from m.

3. SOFT CONSTRAINTS

A soft constraint [7] involves a set of variables and associates a value from a (partially ordered) set to each instantiation of its variables. Such a value is taken from a c-semiring which is defined

by $\langle A, +, \times, 0, 1\rangle$, where A is the set of preference values, $+$ induces an ordering over A (where $a \leq b$ iff $a+b=b$), $\times$ is used to combine preference values, and 0 and 1 are respectively the worst and best element. A Soft Constraint Satisfaction Problem (SCSP) is a tuple $\langle V, D, C, A\rangle$ where V is a set of variables, D is the domain of the variables, C is a set of soft constraints (each one involving a subset of V), A is the set of preference values. An instance of the SCSP framework is obtained by choosing a specific c-semiring. Choosing $S_{FCSP} = \langle [0,1], max, min, 0, 1\rangle$ means that preferences are in [0,1] and we want to maximize the minimum preference. This is the setting of fuzzy CSPs (FCSPs) that we consider in the paper.

An optimal solution of an SCSP is a complete assignment with an undominated preference. Finding an optimal solution is an NP-hard problem, unless certain restrictions are imposed, such as a tree-shaped constraint graph. Constraint propagation may help the search for an optimal solution. Given a variable ordering o, an FCSP is directional arc-consistent (DAC) if, for any two variables x and y linked by a fuzzy constraint, such that x precedes y in the ordering o, we have that, for each a in the domain of x, $f_x(a) = max_{b \in D(y)}(min(f_x(a), f_{xy}(a,b), f_y(b)))$, where f_x, f_y, and f_{xy} are the preference functions of c_x, c_y and c_{xy}. This definition can be generalized to any instance of the SCSP approach by replacing *max* with $+$ and *min* with $\times$. DAC is enough to find the preference level of an optimal solution when the problem has a tree-shaped constraint graph and the variable ordering is compatible with the father-child relation of the tree [7], since the optimum preference level is the best preference level in the domain of the root variable.

4. STABLE MATCHING WITH SOFT CONSTRAINTS

We consider a stable marriage problem with n men and n women, where each man and each woman specify their preferences over the members of the other gender via a set of soft constraints. Each man and woman is described by a set of features, that are represented by the variables of the soft constraint problems. If each variable has d possible values, the number of variables, say f, of each soft constraint problem is $O(log_d n)$.

GS operations. In the GS algorithm, men make proposals, starting from their most preferred woman and going down in their ordering, while women receive proposals and compare these against the man to whom they are currently engaged. We will now model the GS operations.

$Opt(pref(m))$ must return the optimal solution of an SCSP defining the preferences of man m over the women. In general, finding the optimal solution of an SCSP is a computationally difficult problem. However, if the SCSP has a tree-like shape, or a bounded tree-width, it can be done in polynomial time [7].

$Compare(pref(w), m_1, m_2)$ compares two complete assignments m_1 and m_2 and check if m_1 is strictly more preferred to m_2. In SCSPs, this is computationally easy to do, if the combination operator is polynomially computable and there is a polynomial number of constraints. In fact, m_1 is strictly preferred to m_2 when the preference of m_1 for w is strictly greater than that of m_2 for w. Notice that women need only to perform Compare operations. Thus we do not need any restriction on the shape of the constraint graph for women's preferences to make Compare polynomial.

For the $Next(pref(m), w)$ operation, we need to understand how to linearize the solution ordering of an SCSP. In fact, this operation is used to find the next most preferred woman in a man's preference ordering, so when two or more women are tied, we need to put an order over them to understand who to propose first.

Linearizations. We aim to define linearizations where finding the next best solution (that is, applying operation *Next*) is tractable and where solutions which are less distant from optimal ones appear earlier in the linearized order. We define three linearizations L_1, L_2, and L_3 which break ties by taking into account the distance of a solution preference from the optimal preference (L_1), or also the minimum number of preference values for parts of the solutions to be changed to make the solution optimal (L_2), or also the amount of change required (L_3). Among the solutions which are still in a tie, we put first those that are lexicographically earlier, according to an ordering over variables and domain values.

We will now see how to perform operation *Next* on such three linearizations. We will call these operations $Next_i$, for $L_i = 1, 2, 3$. From results in [2], we know that performing $Next_1$ can be accomplished in polynomial time when we have a tree-like fuzzy CSPs. To perform $Next_2$ and $Next_3$ for tree-shaped fuzzy CSPs, when we already have the top $k-1$ solutions, we find the top k solution according L_2 and L_3 by computing the top k solutions of a set of weighted CSPs with a bounded tree-width. This is polynomial [3].

Our algorithm, which we call *KCheapest*, works, with suitable variants, for both L_2 and L_3. The input is a tree-shaped fuzzy CSP P, an integer k, and a linearization L (either L_2 or L_3). The output is a set of top k solutions of P according to the linearization. *KCheapest* returns the top k solutions of P according to L_2 and L_3 in polynomial time, when k is polynomial in f and d. As stated before, from [6] we know that on average only 2% of all proposals are made, so the idea is to call *KCheapest* with $k = 2\%$ of the maximum number of proposals and cache the returned set of solutions. Only when all cached solutions have already been returned, we need to call *KCheapest* again.

If we run the GS algorithm on any of the linearizations we defined, by definition we obtain a matching which is stable w.r.t. this linearization. When we use soft constraints with a bounded tree-width for the men, each proposal in the GS algorithm takes $O(poly(f))$ time, where f is the number of variables and each Compare operation takes $O(poly(m))$, where m is the number of soft constraints. So, overall, the GS algorithm may need up to $O(n^2 \times poly(f) \times poly(m))$ time, although the number of proposals have been shown to be much lower in practice [6].

5. REFERENCES

[1] C. Boutilier, R. I. Brafman, C. Domshlak, H. H. Hoos, and D. Poole. CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. *JAIR*, 21:135–191, 2004.

[2] R. I. Brafman, F. Rossi, D. Salvagnin, K. B. Venable, and T. Walsh. Finding the next solution in constraint- and preference-based knowledge representation formalisms. In *Proc. KR 2010*, 2010.

[3] R. Dechter, N. Flerova, and R. Marinescu. Search algorithms for m best solutions for graphical models. In *Proc. AAAI 2012*, 2012.

[4] D. Gale and L. S. Shapley. College admissions and the stability of marriage. *Amer. Math. Monthly*, 69:9–14, 1962.

[5] D. Gusfield and R. W. Irving. *The Stable Marriage Problem: Structure and Algorithms*. MIT Press, Boston, Mass., 1989.

[6] E. Pilotto, F. Rossi, K. B. Venable, and T. Walsh. Compact preference representation in stable marriage problems. In *Proc. ADT 2009*, pages 390–401, 2009.

[7] F. Rossi, P. V. Beek, and T. Walsh. *Handbook of Constraint Programming*. Elsevier, 2006.