Notes for the Course "Autonomous Agents and Multiagent Systems" 2017/2018

Francesco Amigoni

Current address: Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milano, Italy
E-mail address: francesco.amigoni@polimi.it

Last update: November 10, 2017.

CHAPTER 1

Introduction

This document collects some notes that inegrate the content of the textbook [**4**] for the course "Autonomous Agents and Multiagent Systems" at the Politecnico di Milano. It is intended to supply some additional algorithms and concepts (when possible, retaining the same notation of the textbook). References to the textbook will be in the form [Chapter x, Section y].

CHAPTER 2

Negotiation

[Chapter 4, Section 3]

A protocol more general than the Rubinstein's alternating offers protocol, in the sense that it could be applied to settings with arbitrary set of outcomes O and with arbitrary utility functions is the *monotonic concession protocol*.

It assumes:

- An arbitrary set of outcomes O. Each outcome is denoted by $x \in O$.
- Two agents, called a and b, with arbitray utility functions $U^a : O \to \mathbb{R}$ and $U^b : O \to \mathbb{R}$.
- The negotiation proceeds in rounds, according to time steps $t = 1, 2, \ldots$. At each round, both agents simultaneously make an offer (proposal) and decide if the other's proposal is accepted or rejected. If at least one agent accepts, then the agreement is reached, otherwise the negotiation proceeds to the next round.

Algorithm for *monotonic concession* is reported as Algorithm 1. The reported algorithm is for agent a; a similar algorithm can be defined for agent b.

A possible strategy in the else statement of line 4 is to select a new x such that $U^b(x) = U^b(x^{(a)}) + \epsilon$, for a given ϵ, namely to concede a fixed ϵ to the opponent at each round.

Some properties of the monotonic concession protocol are:

- The protocol is easily verifiable: both agents can see if rules are obeyed.
- The convergence to an agreement can be slow. Convergence speed depends on the size of O, on ϵ, and on the utility functions of the agents.
- Agents should know others' utility functions. This assumption could be irrealistic.
- If both agents accept at the same round and accepted offers $x^{(a)}$ and $x^{(b)}$ are different, then a tie breaking mechanism is used (e.g., random selection of an offer).

A more complex strategy for deciding the amount of concession in line 4 of the above algorithm is the *Zeuthen strategy* (Algorithm 2).

Algorithm 1 Monotonic concession

(1) $x^{(a)} \leftarrow \text{argmax}_{x \in O} U^a(x)$
(2) propose offer $x^{(a)}$
(3) receive proposed offer $x^{(b)}$
(4) **if** $U^a(x^{(b)}) \geq U^a(x^{(a)})$, **then** accept $x^{(b)}$ and **return**, **else** $x^{(a)} \leftarrow x \in O$ such that $U^b(x) \geq U^b(x^{(a)})$ (and $U^a(x) \geq 0$)
(5) **goto** 2

Algorithm 2 Zeuthen strategy

(1) $x^{(a)} \leftarrow \text{argmax}_{x \in O} U^a(x)$
(2) propose offer $x^{(a)}$
(3) receive proposed offer $x^{(b)}$
(4) **if** $U^a(x^{(b)}) \geq U^a(x^{(a)})$, **then** accept $x^{(b)}$ and **return**
(5) $risk_a \leftarrow \frac{U^a(x^{(a)}) - U^a(x^{(b)})}{U^a(x^{(a)})}$
(6) $risk_b \leftarrow \frac{U^b(x^{(b)}) - U^b(x^{(a)})}{U^b(x^{(b)})}$
(7) **if** $risk_a < risk_b$, **then** $x^{(a)} \leftarrow x \in O$ such that $risk_a > risk_b$ when considering x instead of $x^{(a)}$ and **goto** 2
(8) **goto** 3

Zeuthen strategy terminates with an agreement that is individually rational and Pareto optimal. Even more, the agreement reached is a Nash bargaining solution. Also Zeuthen strategy requires the agents to know each other's utility functions but usually converges much more quickly than basic monotonic concession.

CHAPTER 3

Auctions

An algorithm for solving the winner determination problem in combinatorial auctions as formulated in [Chapter 7, Section 7] is based on building a tree structure representing all the possible allocations starting from the sets S of goods for which a bid has been received, such that each allocation is composed of disjoint sets and the sets of an allocation contain all the goods in G. The tree is built by using the following rule: the children of a node are the sets S of goods for which a bid has been received such that (1) they include the smallest good in G that is not present along the path from the root to the node and (2) they do not include any good already present on the path. An example is shown in Figure 3.0.1 for the bids over goods $G = \{1, 2, 3, 4, 5\}$ reported in Table 1. Note that, in the table, only the largest bid $\overline{v}(S)$ is considered for each S and two "dummy bids" have been added for single goods $\{3\}$ and $\{4\}$ which received no "actual" bids. In the tree, each path from the root to a leaf is a possibile allocation composed of sets S of goods (apart from the names of the agents to which the sets S are allocated) and can be associated to its value g, namely, to the revenue for the auctioneer, calculated as the sum of $\overline{v}(S)$ for all S in the allocation. For example, the lestmost path in Figure 3.0.1 has $g = 6 + 2 + 0 = 8$, while the rightmost path in the same figure has $g = 5 + 4 + 0 + 0 + 1 = 10$. Solving the winner determination problem comes down to select the path with the largest value on the tree.

S	$\overline{v}(S)$
$\{1\}$	5
$\{2\}$	4
$\{3\}$	0
$\{4\}$	0
$\{5\}$	1
$\{1, 2\}$	6
$\{1, 3, 5\}$	7
$\{1, 4\}$	5
$\{2, 5\}$	10
$\{3, 5\}$	2

TABLE 1. An example of received bids.

Algorithm 3 Depth-first branch-and-bound search algorithm

(1) $x* \leftarrow \{\}$
(2) $g* \leftarrow 0$
(3) branch-on-items$(1, \{\})$

branch-on-items(j, x)

(1) **if** sets S in x contain all goods in G
(2) **then**
(3) **if** $\sum_{S \in x} \overline{v}(S) < g*$ **then** $g* \leftarrow \sum_{S \in x} \overline{v}(S)$ and $x* \leftarrow x$ **endif**
(4) **return**
(5) **endif**
(6) **for** sets S such that $j \in S$ and $S \cap (\cup_{S' \in x} S') = \{\}$
(7) $x' \leftarrow x + S$
(8) **if** $\sum_{S \in x'} \overline{v}(S) + h(x') > g*$, **then** branch-on-items(j', x')
(9) **endfor**

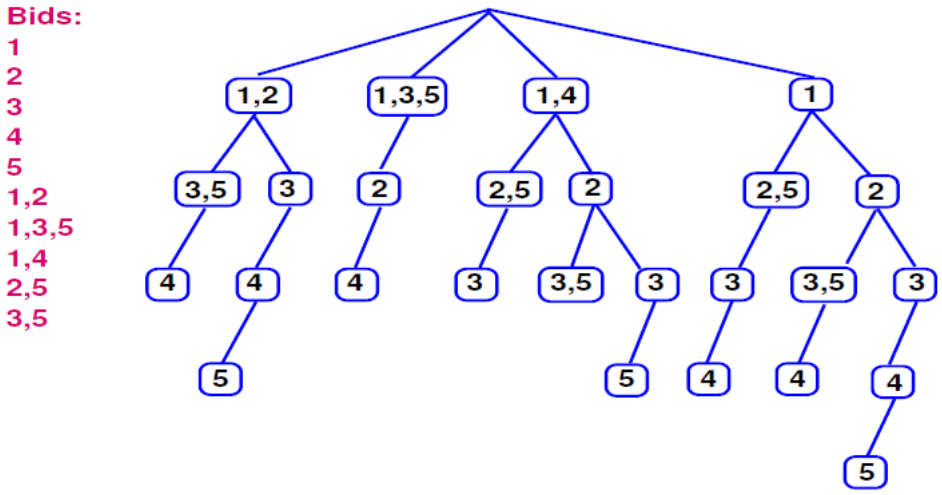

FIGURE 3.0.1. The tree representing all allocations for the bids of Table 1 (from [**2**]).

To efficiently explore the tree to find the best allocation, the *depth-first branch-and-bound search algorithm* introdcued in [**1**] and reported as Algorithm 3 can be used. Global variables $x*$ and $g*$ represent the current best allocation and its value, respectively. Heuristic function $h(x')$ is an upper bound to the value of goods not yet allocated in x'. A possible heuristic function is

$$h(x') = \sum_{j \in \{\text{goods not allocated in } x'\}} \max_{S \text{ such that } j \in S} \frac{\overline{v}(S)}{|S|}.$$

In line 8, j' is the smallest good not allocated in x'.

CHAPTER 4

Coalition formation

The distribued algorithm to (hopefully) find the best coalition structure by Shehory and Kraus [**3**] discussed in [Chapter 8, Section 5.4] is reported as Algorithm4, which refers to the algorithm executed by a generic agent a_i.

Algorithm 4 Shehory and Kraus algorithm

(1) $\mathcal{C}_i \leftarrow$ set of all coalitions that include agent a_i
(2) $C_i^* \leftarrow \text{argmax}_{C \in \mathcal{C}_i} \frac{v(C)}{|C|}$
(3) broadcast (a_i, C_i^*), receive other broadcasts, put received (a_j, C_j^*) in $\mathcal{C}^*$ (including (a_i, C_i^*))
(4) $C_{max} \leftarrow$ largest subset of set of agents A such that, for all $a_j \in C_{max}$, $(a_j, C_{max}) \in \mathcal{C}^*$
(5) **if** $a_i \in C_{max}$, **then** join C_{max} and **return**
(6) delete from $\mathcal{C}_i$ all coalitions that include agents from C_{max}
(7) **if** $\mathcal{C}_i$ is not empty, **then goto** 2
(8) **return**

Bibliography

[1] Fujishima, Y., Leyton-Brown, K., Shoham, Y., "Taming the Computational Complexity of Combinatorial Auctions: Optimal and Approximate Approaches", *Proceedings of the International Joint Conference on Artificial Intelligence*, 1999, p. 548-553.

[2] Sandholm, T., "Optimal Winner Determination Algorithms", in Cramton, P., Shoham, Y., and Steinberg, R. (editors), *Combinatorial Auctions*, The MIT Press, 2006, p. 337-368.

[3] Shehory, O., Kraus S. "Methods for Task Allocation via Agent Coalition Formation", *Artificial Intelligence*, 101(1-2), 1998, p. 165-200.

[4] Weiss, G. (editor), *Multiagent Systems* (2nd edition), The MIT Press, 2013.