Approximate and Online Multi-Issue Negotiation

Shaheen S. Fatima
Department of Computer Science
University of Liverpool
Liverpool L69 3BX, UK.
shaheen@csc.liv.ac.uk

Michael Wooldridge
Department of Computer Science
University of Liverpool
Liverpool L69 3BX, UK.
mjw@csc.liv.ac.uk

Nicholas R. Jennings
School of Electronics and Computer Science
University of Southampton
Southampton SO17 1BJ, UK.
nrj@ecs.soton.ac.uk

ABSTRACT

This paper analyzes bilateral multi-issue negotiation between self-interested autonomous agents. The agents have time constraints in the form of both deadlines and discount factors. There are $m > 1$ issues for negotiation where each issue is viewed as a pie of size one. The issues are "indivisible" (i.e., individual issues cannot be split between the parties; each issue must be allocated in its entirety to either agent). Here different agents value different issues differently. Thus, the problem is for the agents to decide how to allocate the issues between themselves so as to maximize their individual utilities. For such negotiations, we first obtain the equilibrium strategies for the case where the issues for negotiation are known a priori to the parties. Then, we analyse their time complexity and show that finding the equilibrium offers is an NP-hard problem, even in a complete information setting. In order to overcome this computational complexity, we then present negotiation strategies that are *approximately optimal* but computationally efficient, and show that they form an equilibrium. We also analyze the *relative error* (i.e., the difference between the true optimum and the approximate). The time complexity of the approximate equilibrium strategies is $\mathcal{O}(nm/\epsilon^2)$ where n is the negotiation deadline and ϵ the relative error. Finally, we extend the analysis to *online* negotiation where different issues become available at different time points and the agents are uncertain about their valuations for these issues. Specifically, we show that an approximate equilibrium exists for online negotiation and show that the expected difference between the optimum and the approximate is $\mathcal{O}(\sqrt{m})$. These approximate strategies also have polynomial time complexity.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent Systems

General Terms

Algorithms, Design, Theory

Keywords

Game-theory, Negotiation, Approximation, Online Computation.

AAMAS'07 May 14–18 2007, Honolulu, Hawai'i, USA.

1. INTRODUCTION

Negotiation is a key form of interaction in multiagent systems. It is a process in which disputing agents decide how to divide the gains from cooperation. Since this decision is made jointly by the agents themselves [20, 19, 13, 15], each party can only obtain what the other is prepared to allow them. Now, the simplest form of negotiation involves two agents and a single issue. For example, consider a scenario in which a buyer and a seller negotiate on the price of a good. To begin, the two agents are likely to differ on the price at which they believe the trade should take place, but through a process of joint decision-making they either arrive at a price that is mutually acceptable or they fail to reach an agreement. Since agents are likely to begin with different prices, one or both of them must move toward the other, through a series of offers and counter offers, in order to obtain a mutually acceptable outcome. However, before the agents can actually perform such negotiations, they must decide the rules for making offers and counter offers. That is, they must set the negotiation protocol [20]. On the basis of this protocol, each agent chooses its strategy (i.e., what offers it should make during the course of negotiation). Given this context, this work focuses on competitive scenarios with self-interested agents. For such cases, each participant defines its strategy so as to maximise its individual utility.

However, in most bilateral negotiations, the parties involved need to settle more than one issue. For this case, the issues may be *divisible* or *indivisible* [4]. For the former, the problem for the agents is to decide how to split each issue between themselves [21]. For the latter, the individual issues cannot be divided. An issue, in its entirety, must be allocated to either of the two agents. Since the agents value different issues differently, they must come to terms about who will take which issue. To date, most of the existing work on multi-issue negotiation has focussed on the former case [7, 2, 5, 23, 11, 6]. However, in many real-world settings, the issues are indivisible. Hence, our focus here is on negotiation for indivisible issues. Such negotiations are very common in multi-agent systems. For example, consider the case of task allocation between two agents. There is a set of tasks to be carried out and different agents have different preferences for the tasks. The tasks cannot be partitioned; a task must be carried out by one agent. The problem then is for the agents to negotiate about who will carry out which task.

A key problem in the study of multi-issue negotiation is to determine the equilibrium strategies. An equally important problem, especially in the context of software agents, is to find the time complexity of computing the equilibrium offers. However, such computational issues have so far received little attention. As we will show, this is mainly due to the fact that existing work (describe in Section 5) has mostly focused on negotiation for divisible issues

978-81-904262-7-5 (RPS)

and finding the equilibrium for this case is computationally easier than that for the case of indivisible issues. Our primary objective is, therefore, to answer the computational questions for the latter case for the types of situations that are commonly faced by agents in real-world contexts. Thus, we consider negotiations in which there is *incomplete information* and *time constraints*. Incompleteness of information on the part of negotiators is a common feature of most practical negotiations. Also, agents typically have time constraints in the form of both deadlines and discount factors. Deadlines are an essential element since negotiation cannot go on indefinitely, rather it must end within a reasonable time limit. Likewise, discount factors are essential since the goods may be perishable or their value may decline due to inflation. Moreover, the strategic behaviour of agents with deadlines and discount factors differs from those without (see [21] for single issue bargaining without deadlines and [23, 13] for bargaining with deadlines and discount factors in the context of divisible issues).

Given this, we consider indivisible issues and first analyze the strategic behaviour of agents to obtain the equilibrium strategies for the case where all the issues for negotiation are known a priori to both agents. For this case, we show that the problem of finding the equilibrium offers is NP-hard, even in a complete information setting. Then, in order to overcome the problem of time complexity, we present strategies that are *approximately optimal* but computationally efficient, and show that they form an equilibrium. We also analyze the *relative error* (i.e., the difference between the true optimum and the approximate). The time complexity of the approximate equilibrium strategies is $\mathcal{O}(nm/\epsilon^2)$ where n is the negotiation deadline and ϵ the relative error. Finally, we extend the analysis to *online* negotiation where different issues become available at different time points and the agents are uncertain about their valuations for these issues. Specifically, we show that an approximate equilibrium exists for online negotiation and show that the expected difference between the optimum and the approximate is $\mathcal{O}(\sqrt{m})$. These approximate strategies also have polynomial time complexity.

In so doing, our contribution lies in analyzing the computational complexity of the above multi-issue negotiation problem, and finding the approximate and online equilibria. No previous work has determined these equilibria. Since software agents have limited computational resources, our results are especially relevant to such resource bounded agents.

The remainder of the paper is organised as follows. We begin by giving a brief overview of single-issue negotiation in Section 2. In Section 3, we obtain the equilibrium for multi-issue negotiation and show that finding equilibrium offers is an NP-hard problem. We then present an approximate equilibrium and evaluate its approximation error. Section 4 analyzes online multi-issue negotiation. Section 5 discusses the related literature and Section 6 concludes.

2. SINGLE-ISSUE NEGOTIATION

We adopt the single issue model of [27] because this is a model where, during negotiation, the parties are allowed to make offers from a set of discrete offers. Since our focus is on indivisible issues (i.e., parties are allowed to make one of two possible offers: zero or one), our scenario fits in well with [27]. Hence we use this basic single issue model and extend it to multiple issues. Before doing so, we give an overview of this model and its equilibrium strategies.

There are two strategic agents: a and b. Each agent has time constraints in the form of deadlines and discount factors. The two agents negotiate over a single indivisible issue (i). This issue is a 'pie' of size 1 and the agents want to determine who gets the pie. There is a deadline (i.e., a number of rounds by which negotiation must end). Let $n \in \mathbb{N}^+$ denote this deadline. The agents use an alternating offers protocol (as the one of Rubinstein [18]), which proceeds through a series of time periods. One of the agents, say a, starts negotiation in the first time period (i.e., $t = 1$) by making an offer ($x_i = 0$ or 1) to b. Agent b can either accept or reject the offer. If it accepts, negotiation ends in an agreement with a getting x_i and b getting $y_i = 1 - x_i$. Otherwise, negotiation proceeds to the next time period, in which agent b makes a counter-offer. This process of making offers continues until one of the agents either accepts an offer or quits negotiation (resulting in a conflict). Thus, there are three possible actions an agent can take during any time period: accept the last offer, make a new counter-offer, or quit the negotiation.

An essential feature of negotiations involving alternating offers is that the agents' utilities decrease with time [21]. Specifically, the decrease occurs at each step of offer and counteroffer. This decrease is represented with a discount factor denoted $0 < \delta_i \leq 1$ for both[1] agents.

Let $[x_i^t, y_i^t]$ denote the offer made at time period t where x_i^t and y_i^t denote the share for agent a and b respectively. Then, for a given pie, the set of possible offers is:

$$\{[x_i^t, y_i^t] : x_i^t = 0 \text{ or } 1,\ y_i^t = 0 \text{ or } 1,\ \ and\ \ x_i^t + y_i^t = 1\}$$

At time t, if a and b receive a share of x_i^t and y_i^t respectively, then their utilities are:

$$u_i^a(x_i^t, t) = \begin{cases} x_i^t \times \delta^{t-1} & \text{if } t \leq n \\ 0 & \text{otherwise} \end{cases}$$

$$u_i^b(y_i^t, t) = \begin{cases} y_i^t \times \delta^{t-1} & \text{if } t \leq n \\ 0 & \text{otherwise} \end{cases}$$

The conflict utility (i.e., the utility received in the event that no deal is struck) is zero for both agents.

For the above setting, the agents reason as follows in order to determine what to offer at $t = 1$. We let $\text{A}(1)$ ($\text{B}(1)$) denote a's (b's) equilibrium offer for the first time period. Let agent a denote the first mover (i.e., at $t = 1$, a proposes to b who should get the pie). To begin, consider the case where the deadline for both agents is $n = 1$. If b accepts, the division occurs as agreed; if not, neither agent gets anything (since $n = 1$ is the deadline). Here, a is in a powerful position and is able to propose to keep 100 percent of the pie and give nothing to b [2]. Since the deadline is $n = 1$, b accepts this offer and agreement takes place in the first time period.

Now, consider the case where the deadline is $n = 2$. In order to decide what to offer in the first round, a looks ahead to $t = 2$ and reasons backwards. Agent a reasons that if negotiation proceeds to the second round, b will take 100 percent of the pie by offering $[0, 1]$ and leave nothing for a. Thus, in the first time period, if a offers b anything less than the whole pie, b will reject the offer. Hence, during the first time period, agent a offers $[0, 1]$. Agent b accepts this and an agreement occurs in the first time period.

In general, if the deadline is n, negotiation proceeds as follows. As before, agent a decides what to offer in the first round by looking ahead as far as $t = n$ and then reasoning backwards. Agent a's

[1]Having a different discount factor for different agents only makes the presentation more involved without leading to any changes in the analysis of the strategic behaviour of the agents or the time complexity of finding the equilibrium offers. Hence we have a single discount factor for both agents.

[2]It is possible that b may reject such a proposal. However, irrespective of whether b accepts or rejects the proposal, it gets zero utility (because the deadline is $n = 1$). Thus, we assume that b accepts a's offer.

offer for $t = 1$ depends on who the offering agent is for the last time period. This, in turn, depends on whether n is odd or even. Since a makes an offer at $t = 1$ and the agents use the alternating offers protocol, the offering agent for the last time period is b if n is even and it is a if n is odd. Thus, depending on whether n is odd or even, a makes the following offer at $t = 1$:

$$\text{A}(1) = \begin{cases} \text{OFFER } [1,0] & \text{IF ODD } n \\ \text{ACCEPT} & \text{IF } b\text{'s TURN} \end{cases}$$

$$\text{B}(1) = \begin{cases} \text{OFFER } [0,1] & \text{IF EVEN } n \\ \text{ACCEPT} & \text{IF } a\text{'s TURN} \end{cases}$$

Agent b accepts this offer and negotiation ends in the first time period. Note that the equilibrium outcome depends on who makes the first move. Since we have two agents and either of them could move first, we get two possible equilibrium outcomes.

On the basis of the above equilibrium for single-issue negotiation with complete information, we first obtain the equilibrium for multiple issues and then show that computing these offers is a hard problem. We then present a time efficient approximate equilibrium.

3. MULTI-ISSUE NEGOTIATION

We first analyse the complete information setting. This section forms the base which we extend to the case of information uncertainty in Section 4.

Here a and b negotiate over $m > 1$ indivisible issues. These issues are m distinct pies and the agents want to determine how to distribute the pies between themselves. Let $S = \{1, 2, \ldots, m\}$ denote the set of m pies. As before, each pie is of size 1. Let the discount factor for issue c, where $1 \leq c \leq m$, be $0 < \delta_c \leq 1$. For each issue, let n denote each agent's deadline. In the offer for time period t (where $1 \leq t \leq n$), agent a's (b's) share for each of the m issues is now represented as an m element vector $x^t \in \mathbb{B}^m$ ($y^t \in \mathbb{B}^m$) where $\mathbb{B}$ denotes the set $\{0, 1\}$. Thus, if agent a's share for issue c at time t is x_c^t, then agent b's share is $y_c^t = (1 - x_c^t)$. The shares for a and b are together represented as the package $[x^t, y^t]$.

As is traditional in multi-issue utility theory, we define an agent's cumulative utility using the standard additive form [12]. The functions $U^a : \mathbb{B}^m \times \mathbb{B}^m \times \mathbb{N}^+ \rightarrow \mathbb{R}$ and $U^b : \mathbb{B}^m \times \mathbb{B}^m \times \mathbb{N}^+ \rightarrow \mathbb{R}$ give the cumulative utilities for a and b respectively at time t. These are defined as follows:

$$U^a([x^t, y^t], t) = \begin{cases} \Sigma_{c=1}^m k_c^a u_c^a(x_c^t, t) & \text{if } t \leq n \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

$$U^b([x^t, y^t], t) = \begin{cases} \Sigma_{c=1}^m k_c^b u_c^b(y_c^t, t) & \text{if } t \leq n \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where $k^a \in \mathbb{N}_+^m$ denotes an m element vector of constants for agent a and $k^b \in \mathbb{N}_+^m$ that for b. Here $\mathbb{N}_+$ denotes the set of positive integers. These vectors indicate how the agents value different issues. For example, if $k_c^a > k_{c+1}^a$, then agent a values issue c more than issue $c + 1$. Likewise for agent b. In other words, the m issues are *perfect substitutes* (i.e., all that matters to an agent is its total utility for all the m issues and not that for any subset of them). In all the settings we study, the issues will be perfect substitutes. To begin each agent has complete information about all negotiation parameters (i.e., n, m, k_c^a, k_c^b, and δ_c for $1 \leq c \leq m$).

Now, multi-issue negotiation can be done using different procedures. Broadly speaking, there are three key procedures for negotiating multiple issues [19]:

1. the *package deal procedure* where all the issues are settled together as a bundle,
2. the *sequential procedure* where the issues are discussed one after another, and
3. the *simultaneous procedure* where the issues are discussed in parallel.

Between these three procedures, the package deal is known to generate Pareto optimal outcomes [19, 6]. Hence we adopt it here. We first give a brief description of the procedure and then determine the equilibrium strategies for it.

3.1 The package deal procedure

In this procedure, the agents use the same protocol as for single-issue negotiation (described in Section 2). However, an offer for the package deal includes a proposal for each issue under negotiation. Thus, for m issues, an offer includes m divisions, one for each issue. Agents are allowed to either accept a complete offer (i.e., all m issues) or reject a complete offer. An agreement can therefore take place either on all m issues or on none of them.

As per the single-issue negotiation, an agent decides what to offer by looking ahead and reasoning backwards. However, since an offer for the package deal includes a share for all the m issues, the agents can now make tradeoffs across the issues in order to maximise their cumulative utilities.

For $1 \leq c \leq m$, the equilibrium offer for issue c at time t is denoted as $[a_c^t, b_c^t]$ where a_c^t and b_c^t denote the shares for agent a and b respectively. We denote the equilibrium package at time t as $[a^t, b^t]$ where $a^t \in \mathbb{B}^m$ ($b^t \in \mathbb{B}^m$) is an m element vector that denotes a's (b's) share for each of the m issues. Also, for $1 \leq c \leq m$, δ_c is the discount factor for issue c. The symbols $\mathbf{0}$ and $\mathbf{1}$ denote m element vectors of zeroes and ones respectively. Note that for $1 \leq t \leq n$, $a_c^t + b_c^t = 1$ (i.e., the sum of the agents' shares (at time t) for each pie is one). Finally, for time period t (for $1 \leq t \leq n$) we let $\text{A}(t)$ (respectively $\text{B}(t)$) denote the equilibrium strategy for agent a (respectively b).

3.2 Equilibrium strategies

As mentioned in Section 1, the package deal allows agents to make tradeoffs. We let TRADEOFFA (TRADEOFFB) denote agent a's (b's) function for making tradeoffs. We let P denote a set of parameters to the procedure TRADEOFFA (TRADEOFFB) where $P = \{k^a, k^b, \delta, m\}$. Given this, the following theorem characterises the equilibrium for the package deal procedure.

THEOREM 1. *For the package deal procedure, the following strategies form a Nash equilibrium. The equilibrium strategy for $t = n$ is:*

$$\text{A}(n) = \begin{cases} \text{OFFER } [\mathbf{1}, \mathbf{0}] & \text{IF } a\text{'s TURN} \\ \text{ACCEPT} & \text{IF } b\text{'s TURN} \end{cases}$$

$$\text{B}(n) = \begin{cases} \text{OFFER } [\mathbf{0}, \mathbf{1}] & \text{IF } b\text{'s TURN} \\ \text{ACCEPT} & \text{IF } a\text{'s TURN} \end{cases}$$

For all preceding time periods $t < n$, if $[x^t, y^t]$ denotes the offer made at time t, then the equilibrium strategies are defined as follows:

$$\text{A}(t) = \begin{cases} \text{OFFER TRADEOFFA}(P, \text{UB}(t), t) & \text{IF } a\text{'s TURN} \\ \textit{If } (U^a([x^t, y^t], t) \geq \text{UA}(t)) \text{ ACCEPT} & \\ \textit{else REJECT} & \text{IF } b\text{'s TURN} \end{cases}$$

$$\text{B}(t) = \begin{cases} \text{OFFER TRADEOFFB}(P, \text{UA}(t), t) & \text{IF } b\text{'s TURN} \\ \textit{If } (U^b([x^t, y^t], t) \geq \text{UB}(t)) \text{ ACCEPT} & \\ \textit{else REJECT} & \text{IF } a\text{'s TURN} \end{cases}$$

where $\text{UA}(t) = U^a([a^{t+1}, b^{t+1}], t+1)$ *and* $\text{UB}(t) = U^b([a^{t+1}, b^{t+1}], t+1)$.

PROOF. *We look ahead to the last time period (i.e., $t = n$) and then reason backwards. To begin, if negotiation reaches the deadline (n), then the agent whose turn it is takes everything and leaves nothing for its opponent. Hence, we get the strategies* $\text{A}(n)$ *and* $\text{B}(n)$ *as given in the statement of the theorem.*

In all the preceding time periods ($t < n$), the offering agent proposes a package that gives its opponent a cumulative utility equal to what the opponent would get from its own equilibrium offer for the next time period. During time period t, either a or b could be the offering agent. Consider the case where a makes an offer at t. The package that a offers at t gives b a cumulative utility of $U^b([a^{t+1}, b^{t+1}], t+1)$. However, since there is more than one issue, there is more than one package that gives b this cumulative utility. From among these packages, a offers the one that maximises its own cumulative utility (because it is a utility maximiser). Thus, the problem for a is to find the package $[a^t, b^t]$ so as to:

$$\text{maximize} \sum_{c=1}^{m} k_c^a (1 - b_c^t)\delta_c^{t-1} \quad (3)$$

$$\text{such that} \sum_{c=1}^{m} b_c^t k_c^b \geq \text{UB}(t)$$

$$b_c^t = 0 \text{ or } 1 \qquad \text{for } 1 \leq c \leq m$$

where $\text{UB}(t)$, δ_c^{t-1}, k_c^a, *and* k_c^b *are constants and* b_c^t *($1 \leq c \leq m$) is a variable.*

Assume that the function TRADEOFFA *takes parameters P,* $\text{UB}(t)$, *and t, to solve the maximisation problem given in Equation 3 and returns the corresponding package. If there is more than one package that solves Equation 3, then* TRADEOFFA *returns any one of them (because agent a gets equal utility from all such packages and so does agent b). The function* TRADEOFFB *for agent b is analogous to that for a.*

On the other hand, the equilibrium strategy for the agent that receives an offer is as follows. For time period t, let b denote the receiving agent. Then, b accepts $[x^t, y^t]$ if $\text{UB}(t) \leq U^b([x^t, y^t], t)$, *otherwise it rejects the offer because it can get a higher utility in the next time period. The equilibrium strategy for a as receiving agent is defined analogously.*

In this way, we reason backwards and obtain the offers for the first time period. Thus, we get the equilibrium strategies ($\text{A}(t)$ *and* $\text{B}(t)$*) given in the statement of the theorem.* □

The following example illustrates how the agents make tradeoffs using the above equilibrium strategies.

EXAMPLE 1. *Assume there are $m = 2$ issues for negotiation, the deadline for both issues is $n = 2$, and the discount factor for both issues for both agents is $\delta = 1/2$. Let $k_1^a = 3$, $k_2^a = 1$, $k_1^b = 1$, and $k_2^b = 5$. Let agent a be the first mover. By using backward reasoning, a knows that if negotiation reaches the second time period (which is the deadline), then b will get a hundred percent of both the issues. This gives b a cumulative utility of* $\text{UB}(2) = 1/2 + 5/2 = 3$. *Thus, in the first time period, if b gets anything less than a utility of 3, it will reject a's offer. So, at $t = 1$, a offers the package where it gets issue 1 and b gets issue 2. This gives a cumulative utility of 3 to a and 5 to b. Agent b accepts the package and an agreement takes place in the first time period.*

The maximization problem in Equation 3 can be viewed as the 0-1 knapsack problem[3]. In the 0-1 knapsack problem, we have a set of m *items* where each item has a *profit* and a *weight*. There is a knapsack with a given *capacity*. The objective is to fill the knapsack with items so as to maximize the cumulative profit of the items in the knapsack. This problem is analogous to the negotiation problem we want to solve (i.e., the maximization problem of Equation 3). Since k_c^a and δ_c^{t-1} are constants, maximizing $\sum_{c=1}^{m} k_c^a(1-b_c^t)\delta_c^{t-1}$ is the same as minimizing $\sum_{c=1}^{m} k_c^a b_c^t$. Hence Equation 3 can be written as:

$$\text{minimize} \sum_{c=1}^{m} k_c^a b_c^t \quad (4)$$

$$\text{such that} \sum_{c=1}^{m} b_c^t k_c^b \geq \text{UB}(t)$$

$$b_c^t = 0 \text{ or } 1 \qquad \text{for } 1 \leq c \leq m$$

Equation 4 is a minimization version of the standard 0-1 knapsack problem[4] with m items where k_c^a represents the profit for item c, k_c^b the weight for item c, and $\text{UB}(t)$ the knapsack capacity.

Example 1 was for two issues and so it was easy to find the equilibrium offers. But, in general, it is not computationally easy to find the equilibrium offers of Theorem 1. The following theorem proves this.

THEOREM 2. *For the package deal procedure, the problem of finding the equilibrium offers given in Theorem 1 is NP-hard.*

PROOF. *Finding the equilibrium offers given in Theorem 1 requires solving the 0-1 knapsack problem given in Equation 4. Since the 0-1 knapsack problem is NP-hard [17], the problem of finding equilibrium for the package deal is also NP-hard.* □

3.3 Approximate equilibrium

Researchers in the area of algorithms have found time efficient methods for computing approximate solutions to 0-1 knapsack problems [10]. Hence we use these methods to find a solution to our negotiation problem. At this stage, we would like to point out the main difference between solving the 0-1 knapsack problem and solving our negotiation problem. The 0-1 knapsack problem involves decision making by a single agent regarding which items to place in the knapsack. On the other hand, our negotiation problem involves two players and they are both strategic. Hence, in our case, it is not enough to just find an approximate solution to the knapsack problem, we must also show that such an approximation forms an *equilibrium*.

The traditional approach for overcoming the computational complexity in finding an equilibrium has been to use an *approximate equilibrium* (see [14, 26] for example). In this approach, a strategy profile is said to form an ϵ approximate Nash equilibrium if neither agent can gain more than the constant ϵ by deviating. Hence, our aim is to use the solution to the 0-1 knapsack problem proposed in [10] and show that it forms an approximate equilibrium to our negotiation problem. Before doing so, we give a brief overview of the key ideas that underlie approximation algorithms.

There are two key issues in the design of approximate algorithms [1]:

[3]Note that for the case of divisible issues this is the fractional knapsack problem. The factional knapsack problem is computationally easy; it can be solved in time polynomial in the number of items in the knapsack problem [17]. In contrast, the 0-1 knapsack problem is computationally hard.

[4]Note that for the standard 0-1 knapsack problem the weights, profits and the capacity are positive integers. However a 0-1 knapsack problem with fractions and non positive values can easily be transformed to one with positive integers in time linear in m using the methods given in [8, 17].

1. the quality of their solution, and
2. the time taken to compute the approximation.

The quality of an approximate algorithm is determined by comparing its performance to that of the optimal algorithm and measuring the *relative error* [3, 1]. The relative error is defined as $(z-z^*)/z^*$ where z is the approximate solution and z^* the optimal one. In general, we are interested in finding approximate algorithms whose relative error is bounded from above by a certain constant ϵ, i.e.,

$$(z - z^*)/z^* \quad \leq \quad \epsilon \tag{5}$$

Regarding the second issue of time complexity, we are interested in finding *fully polynomial* approximation algorithms. An approximation algorithm is said to be fully polynomial if for any $\epsilon > 0$ it finds a solution satisfying Equation 5 in time polynomially bounded by size of the problem (for the 0-1 knapsack problem, the problem size is equal to the number of items) and by $1/\epsilon$ [1].

For the 0-1 knapsack problem, Ibarra and Kim [10] presented a fully polynomial approximation method. This method is based on *dynamic programming*. It is a parametric method that takes ϵ as a parameter and for any $\epsilon > 0$, finds a heuristic solution z with relative error at most ϵ, such that the time and space complexity grow polynomially with the number of items m and $1/\epsilon$. More specifically, the space and time complexity are both $\mathcal{O}(m/\epsilon^2)$ and hence polynomial in m and $1/\epsilon$ (see [10] for the detailed approximation algorithm and proof of time and space complexity).

Since the Ibarra and Kim method is fully polynomial, we use it to solve our negotiation problem. This is done as follows. For agent a, let APRX-TRADEOFFA$(P, \text{UB}(t), t, \epsilon)$ denote a procedure that returns an ϵ approximate solution to Equation 4 using the Ibarra and Kim method. The procedure APRX-TRADEOFFB$(P, \text{UA}(t), t, \epsilon)$ for agent b is analogous.

For $1 \leq c \leq m$, the approximate equilibrium offer for issue c at time t is denoted as $[\bar{a}_c^t, \bar{b}_c^t]$ where $\bar{a}_c^t$ and $\bar{b}_c^t$ denote the shares for agent a and b respectively. We denote the equilibrium package at time t as $[\bar{a}^t, \bar{b}^t]$ where $\bar{a}^t \in \mathbb{B}^m$ ($\bar{b}^t \in \mathbb{B}^m$) is an m element vector that denotes a's (b's) share for each of the m issues. Also, as before, for $1 \leq c \leq m$, δ_c is the discount factor for issue c. Note that for $1 \leq t \leq n$, $\bar{a}_c^t + \bar{b}_c^t = 1$ (i.e., the sum of the agents' shares (at time t) for each pie is one). Finally, for time period t (for $1 \leq t \leq n$) we let $\bar{\text{A}}(t)$ (respectively $\bar{\text{B}}(t)$) denote the approximate equilibrium strategy for agent a (respectively b).The following theorem uses this notation and characterizes an approximate equilibrium for multi-issue negotiation.

THEOREM 3. *For the package deal procedure, the following strategies form an ϵ approximate Nash equilibrium. The equilibrium strategy for $t = n$ is:*

$$\bar{\text{A}}(n) = \begin{cases} \text{OFFER } [\boldsymbol{1}, \boldsymbol{0}] & \text{IF } a\text{'s TURN} \\ \text{ACCEPT} & \text{IF } b\text{'s TURN} \end{cases}$$

$$\bar{\text{B}}(n) = \begin{cases} \text{OFFER } [\boldsymbol{0}, \boldsymbol{1}] & \text{IF } b\text{'s TURN} \\ \text{ACCEPT} & \text{IF } a\text{'s TURN} \end{cases}$$

For all preceding time periods $t < n$, if $[x^t, y^t]$ denotes the offer made at time t, then the equilibrium strategies are defined as follows:

$$\bar{\text{A}}(t) = \begin{cases} \text{OFFER APRX-TRADEOFFA}(P, \text{UB}(t), t, \epsilon) & \text{IF } a\text{'s TURN} \\ \textit{If } (U^a([x^t, y^t], t) \geq \text{UA}(t)) \text{ ACCEPT} & \\ \textit{else } \text{REJECT} & \text{IF } b\text{'s TURN} \end{cases}$$

$$\bar{\text{B}}(t) = \begin{cases} \text{OFFER APRX-TRADEOFFB}(P, \text{UA}(t), t, \epsilon) & \text{IF } b\text{'s TURN} \\ \textit{If } (U^b([x^t, y^t], t) \geq \text{UB}(t)) \text{ ACCEPT} & \\ \textit{else } \text{REJECT} & \text{IF } a\text{'s TURN} \end{cases}$$

where $\text{UA}(t) = U^a([\bar{a}^{t+1}, \bar{b}^{t+1}], t+1)$ *and* $\text{UB}(t) = U^b([\bar{a}^{t+1}, \bar{b}^{t+1}], t+1)$. *An agreement takes place at $t = 1$.*

PROOF. *As in the proof for Theorem 1, we use backward reasoning. We first obtain the strategies for the last time period $t = n$. It is straightforward to get these strategies; the offering agent gets a hundred percent of all the issues.*

Then for $t = n-1$, the offering agent must solve the maximization problem of Equation 4 by substituting $t = n-1$ in it. For agent a (b), this is done by APPROX-TRADEOFFA *(*APPROX-TRADEOFFB*). These two functions are nothing but the Ibarra and Kim's approximation method for solving the 0-1 knapsack problem. These two functions take ϵ as a parameter and use the Ibarra and Kim's approximation method to return a package that approximately maximizes Equation 4. Thus, the relative error for these two functions is the same as that for Ibarra and Kim's method (i.e., it is at most ϵ where ϵ is given in Equation 5).*

Assume that a is the offering agent for $t = n-1$. Agent a must offer a package that gives b a cumulative utility equal to what it would get from its own approximate equilibrium offer for the next time period (i.e., $U^b([\bar{a}^{t+1}, \bar{b}^{t+1}], t+1)$ where $[\bar{a}^{t+1}, \bar{b}^{t+1}]$ is the approximate equilibrium package for the next time period). Recall that for the last time period, the offering agent gets a hundred percent of all the issues. Since a is the offering agent for $t = n-1$ and the agents use the alternating offers protocol, it is b's turn at $t = n$. Thus $U^b([\bar{a}^{t+1}, \bar{b}^{t+1}], t+1)$ is equal to b's cumulative utility from receiving a hundred percent of all the issues. Using this utility as the capacity of the knapsack, a uses APPROX-TRADEOFFA *and obtains the approximate equilibrium package for $t = n-1$. On the other hand, if b is the offering agent at $t = n-1$, it uses* APPROX-TRADEOFFB *to obtain the approximate equilibrium package.*

In the same way for $t < n-1$, the offering agent (say a) uses APPROX-TRADEOFFA *to find an approximate equilibrium package that gives b a utility of $U^b([\bar{a}^{t+1}, \bar{b}^{t+1}], t+1)$. By reasoning backwards, we obtain the offer for time period $t = 1$. If a (b) is the offering agent, it proposes the offer* APPROX-TRADEOFFA$(P, \text{UB}(1), 1, \epsilon)$ *(*APPROX-TRADEOFFB$(P, \text{UA}(1), 1, \epsilon)$*). The receiving agent accepts the offer. This is because the relative error in its cumulative utility from the offer is at most ϵ. An agreement therefore takes place in the first time period.* □

THEOREM 4. *The time complexity of finding the ϵ approximate equilibrium offer for the first time period is $\mathcal{O}(nm/\epsilon^2)$.*

PROOF. *The time complexity of* APPROX-TRADEOFFA *and* APPROX-TRADEOFFB *is the same as the time complexity of the Ibarra and Kim method [10] i.e., $\mathcal{O}(m/\epsilon^2)$). In order to find the equilibrium offer for the first time period using backward reasoning,* APPROX-TRADEOFFA *(or* APPROX- TRADEOFFB*) is invoked n times. Hence the time complexity of finding the ϵ approximate equilibrium offer for the first time period is $\mathcal{O}(nm/\epsilon^2)$.* □

This analysis was done in a complete information setting. However an extension of this analysis to an incomplete information setting where the agents have probability distributions over some uncertain parameter is straightforward, as long as the negotiation is done offline; i.e., the agents know their preference for each individual issue before negotiation begins. For instance, consider the case where different agents have different discount factors, and each agent is uncertain about its opponent's discount factor although it knows its own. This uncertainty is modelled with a probability distribution over the possible values for the opponent's discount factor and having this distribution as common knowledge to the agents. All our analysis for the complete information setting still holds for

this incomplete information setting, except for the fact that an agent must now use the given probability distribution and find its opponent's *expected* utility instead of its actual utility. Hence, instead of analyzing an incomplete information setting for offline negotiation, we focus on online multi-issue negotiation.

4. ONLINE MULTI-ISSUE NEGOTIATION

We now consider a more general and, arguably more realistic, version of multi-issue negotiation, where the agents are uncertain about the issues they will have to negotiate about in future. In this setting, when negotiating an issue, the agents know that they will negotiate more issues in the future, but they are uncertain about the details of those issues. As before, let m be the total number of issues that are up for negotiation. The agents have a probability distribution over the possible values of k_c^a and k_c^b. For $1 \leq c \leq m$ let k_c^a and k_c^b be uniformly distributed over [0,1]. This probability distribution, n, and m are common knowledge to the agents. However, the agents come to know k_c^a and k_c^b only just before negotiation for issue c begins. Once the agents reach an agreement on issue c, it cannot be re-negotiated.

This scenario requires online negotiation since the agents must make decisions about an issue prior to having the information about the future issues [3]. We first give a brief introduction to online problems and then draw an analogy between the *online knapsack problem* and the negotiation problem we want to solve.

In an online problem, data is given to the algorithm incrementally, one unit at a time [3]. The online algorithm must also produce the output incrementally: after seeing i units of input it must output the ith unit of output. Since decisions about the output are made with incomplete knowledge about the entire input, an online algorithm often cannot produce an optimal solution. Such an algorithm can only approximate the performance of the optimal algorithm that sees all the inputs in advance. In the design of online algorithms, the main aim is to achieve a performance that is close to that of the optimal offline algorithm on each input. An online algorithm is said to be *stochastic* if it makes decisions on the basis of the probability distributions for the future inputs. The performance of stochastic online algorithms is assessed in terms of the expected difference between the optimum and the approximate solution (denoted $E[z_m^* - z_m]$ where z_m^* is the optimal and z_m the approximate solution). Note that the subscript m is used to indicate the fact that this difference depends on m.

We now describe the protocol for online negotiation and then obtain an approximate equilibrium. The protocol is defined as follows. Let agent a denote the first mover (since we focus on the package deal procedure, the first mover is the same for all the m issues).

Step 1. For $c = 1$, the agents are given the values of k_c^a and k_c^b. These two values are now common[5] knowledge.

Step 2. The agents settle issue c using the alternating offers protocol described in Section 2. Negotiation for issue c must end within n time periods from the start of negotiation on the issue. If an agreement is not reached within this time, then negotiation fails on this and on all remaining issues.

Step 3. The above steps are repeated for issues $c = 2, 3, \ldots, m$. Negotiation for issue c ($2 \leq c \leq m$) begins in the time period following an agreement on issue $c-1$.

[5]We assume common knowledge because it simplifies exposition. However, if k_c^a (k_c^b) is a's (b's) private knowledge, then our analysis will still hold but now an agent must find its opponent's expected utility on the basis of the p.d.fs for k_c^a and k_c^b.

Thus, during time period t, the problem for the offering agent (say a) is to find the optimal offer for issue c on the basis of k_c^a and k_c^b and the probability distribution for k_i^a and k_i^b ($c < i \leq m$). In order to solve this online negotiation problem we draw analogy with the online knapsack problem. Before doing so, however, we give a brief overview of the online knapsack problem.

In the online knapsack problem, there are m items. The agent must examine the m items one at a time according to the order they are input (i.e., as their profit and size coefficients become known). Hence, the algorithm is required to decide whether or not to include each item in the knapsack as soon as its weight and profit become known, without knowledge concerning the items still to be seen, except for their total number. Note that since the agents have a probability distribution over the weights and profits of the future items, this is a case of stochastic online knapsack problem. Our online negotiation problem is analogous to the online knapsack problem. This analogy is described in detail in the proof for Theorem 5. Again, researchers in algorithms have developed time efficient approximate solutions to the online knapsack problem [16]. Hence we use this solution and show that it forms an equilibrium.

The following theorem characterizes an approximate equilibrium for online negotiation. Here the agents have to choose a strategy without knowing the features of the future issues. Because of this information incompleteness, the relevant equilibrium solution is that of a Bayes' Nash Equilibrium (BNE) in which each agent plays the best response to the other agents with respect to their expected utilities [18]. However, finding an agent's BNE strategy is analogous to solving the online 0-1 knapsack problem. Also, the online knapsack can only be solved approximately [16]. Hence the relevant equilibrium solution concept is approximate BNE (see [26] for example). The following theorem finds this equilibrium using procedures ONLINE- TRADEOFFA and ONLINE-TRADEOFFB which are defined in the proof of the theorem. For a given time period, we let z_m denote the approximately optimal solution generated by ONLINE-TRADEOFFA (or ONLINE-TRADEOFFB) and z_m^* the actual optimum.

THEOREM 5. *For the package deal procedure, the following strategies form an approximate Bayes' Nash equilibrium. The equilibrium strategy for $t = n$ is:*

$$\text{A}(n) = \begin{cases} \text{OFFER } [\boldsymbol{1}, \boldsymbol{0}] & \text{IF } a\text{'s TURN} \\ \text{ACCEPT} & \text{IF } b\text{'s TURN} \end{cases}$$

$$\text{B}(n) = \begin{cases} \text{OFFER } [\boldsymbol{0}, \boldsymbol{1}] & \text{IF } b\text{'s TURN} \\ \text{ACCEPT} & \text{IF } a\text{'s TURN} \end{cases}$$

For all preceding time periods $t < n$, if $[x^t, y^t]$ denotes the offer made at time t, then the equilibrium strategies are defined as follows:

$$\text{A}(t) = \begin{cases} \text{OFFER ONLINE-TRADEOFFA}(P, \text{UB}(t), t) & \text{IF } a\text{'s TURN} \\ \textit{If } (U^a([x^t, y^t], t) \geq \text{UA}(t)) \text{ ACCEPT} & \\ \textit{else REJECT} & \text{IF } b\text{'s TURN} \end{cases}$$

$$\text{B}(t) = \begin{cases} \text{OFFER ONLINE-TRADEOFFB}(P, \text{UA}(t), t) & \text{IF } b\text{'s TURN} \\ \textit{If } (U^b([x^t, y^t], t) \geq \text{UB}(t)) \text{ ACCEPT} & \\ \textit{else REJECT} & \text{IF } a\text{'s TURN} \end{cases}$$

where $\text{UA}(t) = U^a([\bar{a}^{t+1}, \bar{b}^{t+1}], t+1)$ *and* $\text{UB}(t) = U^b([\bar{a}^{t+1}, \bar{b}^{t+1}], t+1)$. *An agreement on issue c takes place at $t = c$. For a given time period, the expected difference between the solution generated by the optimal strategy and that by the approximate strategy is $E[z_m^* - z_m] = \mathcal{O}(\sqrt{m})$.*

PROOF. *As in Theorem 1 we find the equilibrium offer for time period $t = 1$ using backward induction. Let a be the offering agent for $t = 1$ for all the m issues. Consider the last time period $t = n$ (recall from Step 2 of the online protocol that n is the deadline for completing negotiation on the first issue). Since the first mover is the same for all the issues, and the agents make offers alternately, the offering agent for $t = n$ is also the same for all the m issues. Assume that b is the offering agent for $t = n$. As in Section 3, the offering agent for $t = n$ gets a hundred percent of all the m issues. Since b is the offering agent for $t = n$, his utility for this time period is:*

$$\text{UB}(n) \quad = \quad k_1^b \delta_1^{n-1} + 1/2 \sum_{i=2}^{m} \delta_i^{i(n-1)} \tag{6}$$

Recall that k_i^a and k_i^b (for $c < i \leq m$) are not known to the agents. Hence, the agents can only find their expected utilities from the future issues on the basis of the probability distribution functions for k_i^a and k_i^b. However, during the negotiation for issue c the agents know k_c^a but not k_c^b (see Step 1 of the online protocol). Hence, a computes $\text{UB}(n)$ as follows. Agent b's utility from issue $c = 1$ is $k_1^b \delta_1^{n-1}$ (which is the first term of Equation 6). Then, on the basis of the probability distribution functions for k_i^a and k_i^b, agent a computes b's expected utility from each future issue i as $\delta_i^{i(n-1)}/2$ (since k_i^a and k_i^b are uniformly distributed on $[0, 1]$). Thus, b's expected cumulative utility from these $m - c$ issues is $1/2 \sum_{i=2}^{m} \delta_i^{i(n-1)}$ (which is the second term of Equation 6).

Now, in order to decide what to offer for issue $c = 1$, the offering agent for $t = n - 1$ (i.e., agent a) must solve the following online knapsack problem:

$$\begin{aligned} &\textit{maximize} && \Sigma_{i=1}^{m} k_i^a (1 - \bar{b}_i^t) \delta_i^{n-1} \\ &\textit{such that} && \Sigma_{i=1}^{m} k_i^b \bar{b}_i^t \geq \text{UB}(n) \\ & && \bar{b}_i^t = 0 \textit{ or } 1 \qquad \textit{for } 1 \leq i \leq m \end{aligned} \tag{7}$$

The only variables in the above maximization problem are $\bar{b}_i^t$. Now, maximizing $\Sigma_{i=1}^{m} k_i^a (1 - \bar{b}_i^t) \delta_i^{n-1}$ is the same as minimizing $\Sigma_{i=1}^{m} k_i^a \bar{b}_i^t$ since δ_i^{n-1} and k_i^a are constants. Thus, we write Equation 7 as:

$$\begin{aligned} &\textit{minimize} && \Sigma_{i=1}^{m} k_i^a \bar{b}_i^t \\ &\textit{such that} && \Sigma_{i=1}^{m} k_i^b \bar{b}_i^t \geq \text{UB}(n) \\ & && \bar{b}_i^t = 0 \textit{ or } 1 \qquad \textit{for } 1 \leq i \leq m \end{aligned} \tag{8}$$

The above optimization problem is analogous to the online 0-1 knapsack problem. An algorithm to solve the online knapsack problem has already proposed in [16]. This algorithm is called the fixed-choice online algorithm. *It has time complexity linear in the number of items (m) in the knapsack problem. We use this to solve our online negotiation problem. Thus, our* ONLINE-TRADEOFFA *algorithm is nothing but the fixed-choice online algorithm and therefore has the same time complexity as the latter. This algorithm takes the values of k_i^a and k_i^b one at a time and generates an approximate solution to the above knapsack problem. The expected difference between the optimum and approximate solution is $E[z_m^* - z_m] = \mathcal{O}(\sqrt{m})$ [16] (see [16] for the detailed fixed-choice online algorithm and a proof for $E[z_m^* - z_m] = \mathcal{O}(\sqrt{m})$).*

The fixed-choice online algorithm of [16] is a generalization of the basic greedy algorithm for the offline knapsack problem; the idea behind it is as follows. A threshold *value is determined on the basis of the information regarding weights and profits for the 0-1 knapsack problem. The method then includes into the knapsack all items whose profit density (profit density of an item is its profit per unit weight) exceeds the threshold until either the knapsack is filled or all the m items have been considered.*

In more detail, the algorithm ONLINE-TRADEOFFA *works as follows. It first gets the values of k_1^a and k_1^b and finds $\bar{b}_c^t$. Since we have a 0-1 knapsack problem, $\bar{b}_c^t$ can be either zero or one. Now, if $\bar{b}_c^t = 1$ for $t = n$, then $\bar{b}_c^t$ must be one for $1 \leq t < n$ (i.e., a must offer $\bar{b}_c^t = 1$ at $t = 1$). If $\bar{b}_c^t = 1$ for $t = n$, but a offers $\bar{b}_c^t = 0$ at $t = 1$, then agent b gets less utility than what it expects from a's offer and rejects the proposal. Thus, if $\bar{b}_c^t = 1$ for $t = n$, then the optimal strategy for a is to offer $\bar{b}_c^t = 1$ at $t = 1$. Agent b accepts the offer. Thus, negotiation on the first issue starts at $t = 1$ and an agreement on it is also reached at $t = 1$.*

In the next time period (i.e., $t = 2$), negotiation proceeds to the next issue. The deadline for the second issue is n time periods from the start of negotiation on the issue. For $c = 2$, the algorithm ONLINE-TRADEOFFA *is given the values of k_2^a and k_2^b and finds $\bar{b}_c^t$ as described above. Agent offers b_c at $t = 2$ and b accepts. Thus, negotiation on the second issue starts at $t = 2$ and an agreement on it is also reached at $t = 2$.*

This process repeats for the remaining issues $c = 3, \ldots, m$. Thus, each issue is agreed upon in the same time period in which it starts. As negotiation for the next issue starts in the following time period (see step 3 of the online protocol), agreement on issue i occurs at time $t = i$.

On the other hand, if b is the offering agent at $t = 1$, he uses the algorithm ONLINE-TRADEOFFB *which is defined analogously. Thus, irrespective of who makes the first move, all the m issues are settled at time $t = m$.* □

THEOREM 6. *The time complexity of finding the approximate equilibrium offers of Theorem 5 is linear in m.*

PROOF. *The time complexity of* ONLINE-TRADEOFFA *and* ONLINE-TRADEOFFB *is the same as the time complexity of the fixed-choice online algorithm of [16]. Since the latter has time complexity linear in m, the time complexity of* ONLINE-TRADEOFFA *and* ONLINE-TRADEOFFB *is also linear in m.* □

It is worth noting that, for the 0-1 knapsack problem, the lower bound on the expected difference between the optimum and the solution found by any online algorithm is $\Omega(1)$ [16]. Thus, it follows that this lower bound also holds for our negotiation problem.

5. RELATED WORK

Work on multi-issue negotiation can be divided into two main types: that for indivisible issues and that for divisible issues. We first describe the existing work for the case of divisible issues. Since Schelling [24] first noted that the outcome of negotiation depends on the choice of negotiation procedure, much research effort has been devoted to the study of different procedures for negotiating multiple issues. However, most of this work has focussed on the sequential procedure [7, 2]. For this procedure, a key issue is the negotiation agenda. Here the term agenda refers to the order in which the issues are negotiated. The agenda is important because each agent's cumulative utility depends on the agenda; if we change the agenda then these utilities change. Hence, the agents must decide what agenda they will use. Now, the agenda can be decided before negotiating the issues (such an agenda is called exogenous) or it may be decided during the process of negotiation (such an agenda is called endogenous). For instance, Fershtman [7] analyze sequential negotiation with exogenous agenda. A number of researchers have also studied negotiations with an endogenous agenda [2].

In contrast to the above work that mainly deals with sequential negotiation, [6] studies the equilibrium for the package deal procedure. However, all the above mentioned work differs from ours in that we focus on indivisible issues while others focus on the case

where each issue is divisible. Specifically, no previous work has determined an approximate equilibrium for multi-issue negotiation or for online negotiation.

Existing work for the case of indivisible issues has mostly dealt with task allocation problems (for tasks that cannot be partioned) to a group of agents. The problem of task allocation has been previously studied in the context of coalitions involving more than two agents. For example [25] analyze the problem for the case where the agents act so as to maximize the benefit of the system as a whole. In contrast, our focus is on two agents where both of them are self-interested and want to maximize their individual utilities. On the other hand [22] focus on the use of contracts for task allocation to multiple self interested agents but this work concerns finding ways of decommitting contracts (after the initial allocation has been done) so as to improve an agent's utility. In contrast, our focuses on negotiation regarding who will carry out which task.

Finally, online and approximate mechanisms have been studied in the context of auctions [14, 9] but not for bilateral negotiations (which is the focus of our work).

6. CONCLUSIONS

This paper has studied bilateral multi-issue negotiation between self-interested autonomous agents with time constraints. The issues are indivisible and different agents value different issues differently. Thus, the problem is for the agents to decide how to allocate the issues between themselves so as to maximize their individual utilities. Specifically, we first showed that finding the equilibrium offers is an NP-hard problem even in a complete information setting. We then presented approximately optimal negotiation strategies and showed that they form an equilibrium. These strategies have polynomial time complexity. We also analysed the difference between the true optimum and the approximate optimum. Finally, we extended the analysis to online negotiation where the issues become available at different time points and the agents are uncertain about the features of these issues. Specifically, we showed that an approximate equilibrium exists for online negotiation and analysed the approximation error. These approximate strategies also have polynomial time complexity.

There are several interesting directions for future work. First, for online negotiation, we assumed that the constants k_c^a and k_c^b are both uniformly distributed. It will be interesting to analyze the case where k_c^a and k_c^b have other, possibly different, probability distributions. Apart from this, we treated the number of issues as being common knowledge to the agents. In future, it will be interesting to treat the number of issues as uncertain.

7. REFERENCES

[1] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and approximation: Combinatorial optimization problems and their approximability properties*. Springer, 2003.

[2] M. Bac and H. Raff. Issue-by-issue negotiations: the role of information and time preference. *Games and Economic Behavior*, 13:125–134, 1996.

[3] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.

[4] S. J. Brams. *Fair division: from cake cutting to dispute resolution*. Cambridge University Press, 1996.

[5] L. A. Busch and I. J. Horstman. Bargaining frictions, bargaining procedures and implied costs in multiple-issue bargaining. *Economica*, 64:669–680, 1997.

[6] S. S. Fatima, M. Wooldridge, and N. R. Jennings. Multi-issue negotiation with deadlines. *Journal of Artificial Intelligence Research*, 27:381–417, 2006.

[7] C. Fershtman. The importance of the agenda in bargaining. *Games and Economic Behavior*, 2:224–238, 1990.

[8] F. Glover. A multiphase dual algorithm for the zero-one integer programming problem. *Operations Research*, 13:879–919, 1965.

[9] M. T. Hajiaghayi, R. Kleinberg, and D. C. Parkes. Adaptive limited-supply online auctions. In *ACM Conference on Electronic Commerce (ACMEC-04)*, pages 71–80, New York, 2004.

[10] O. H. Ibarra and C. E. Kim. Fast approximation algorithms for the knapsack and sum of subset problems. *Journal of ACM*, 22:463–468, 1975.

[11] R. Inderst. Multi-issue bargaining with endogenous agenda. *Games and Economic Behavior*, 30:64–82, 2000.

[12] R. Keeney and H. Raiffa. *Decisions with Multiple Objectives: Preferences and Value Trade-offs*. New York: John Wiley, 1976.

[13] S. Kraus. *Strategic negotiation in multi-agent environments*. The MIT Press, Cambridge, Massachusetts, 2001.

[14] D. Lehman, L. I. O'Callaghan, and Y. Shoham. Truth revelation in approximately efficient combinatorial auctions. *Journal of the ACM*, 49(5):577–602, 2002.

[15] A. Lomuscio, M. Wooldridge, and N. R. Jennings. A classification scheme for negotiation in electronic commerce. *International Journal of Group Decision and Negotiation*, 12(1):31–56, 2003.

[16] A. Marchetti-Spaccamela and C. Vercellis. Stochastic online knapsack problems. *Mathematical Programming*, 68:73–104, 1995.

[17] S. Martello and P. Toth. *Knapsack problems: Algorithms and computer implementations*. John Wiley and Sons, 1990.

[18] M. J. Osborne and A. Rubinstein. *A Course in Game Theory*. The MIT Press, 1994.

[19] H. Raiffa. *The Art and Science of Negotiation*. Harvard University Press, Cambridge, USA, 1982.

[20] J. S. Rosenschein and G. Zlotkin. *Rules of Encounter*. MIT Press, 1994.

[21] A. Rubinstein. Perfect equilibrium in a bargaining model. *Econometrica*, 50(1):97–109, January 1982.

[22] T. Sandholm and V. Lesser. Levelled commitment contracts and strategic breach. *Games and Economic Behavior: Special Issue on AI and Economics*, 35:212–270, 2001.

[23] T. Sandholm and N. Vulkan. Bargaining with deadlines. In *AAAI-99*, pages 44–51, Orlando, FL, 1999.

[24] T. C. Schelling. An essay on bargaining. *American Economic Review*, 46:281–306, 1956.

[25] O. Shehory and S. Kraus. Methods for task allocation via agent coalition formation. *Artificial Intelligence Journal*, 101(1-2):165–200, 1998.

[26] S. Singh, V. Soni, and M. Wellman. Computing approximate Bayes Nash equilibria in tree games of incomplete information. In *Proceedings of the ACM Conference on Electronic Commerce ACM-EC*, pages 81–90, New York, May 2004.

[27] I. Stahl. *Bargaining Theory*. Economics Research Institute, Stockholm School of Economics, Stockholm, 1972.