This study presents Bargaining Chips: a framework for one-to-
many concurrent composite negotiations, where multiple deals can 
be reached and combined. Our framework is designed to mirror the 
salient aspects of real-life procurement and trading scenarios, in 
which a buyer seeks to acquire a number of items from different 
sellers at the same time. To do so, the buyer needs to successfully 
perform multiple concurrent bilateral negotiations as well as coor-
dinate the composite outcome resulting from each interdependent 
negotiation. This paper contributes to the state of the art by: (1) 
presenting a model and test-bed for addressing such challenges; (2) 
by proposing a new, asynchronous interaction protocol for co-
ordinating concurrent negotiation threads; and (3) by providing 
classes of multi-deal coordinators that are able to navigate this new 
one-to-many multi-deal setting. We show that Bargaining Chips 
can be used to evaluate general asynchronous negotiation and co-
ordination strategies in a setting that generalizes over a number of 
existing negotiation approaches.

CCS CONCEPTS
• Applied computing → Electronic commerce; Multi-criterion opti-
mization and decision-making.

KEYWORDS
One-to-many negotiations, multi-deal, composite negotiations, con-
current negotiations, procurement, asynchronous offers, coordina-
tion

Figure 1: One-to-many concurrent composite negotiations.

1 INTRODUCTION

As businesses are increasingly moving their purchases online, pro-
curement technologies are rapidly emerging and evolving. Digital 
procurement solutions can help automate manual tasks typically 
performed by humans, such as peer-to-peer negotiations between 
buyers and sellers, increasing efficiency and reducing errors and 
risks in the process [9].

In a typical procurement setting, a buyer has a list of goods (or 
products/services) they are interested in, and various sellers may 
have these goods on offer (see Fig. 1). To achieve agreement, the 
buyer negotiates with each seller over the price and quantities of 
a bundle of goods. Often, no single seller can provide all desired 
goods at an affordable price. This creates the need for multiple deals 
over possibly overlapping products, each with a different seller. In 
the example of Fig. 1, our buyer, Bob, wants to purchase multiple 
red, green, and blue chips and must therefore strike deals with three 
different sellers to fulfill his demand appropriately.

A major research challenge for the automation of procurement 
lies in managing concurrent and interdependent negotiations, where 
multiple such deals can be achieved (or fall through) with multiple 
sellers, which all have to be coordinated on the fly. Early work on 
reallocation by distributed negotiation has investigated the sys-
tem convergence to global or local optima, but did not address the 
involvement of strategic agents [4].

In contrast, we focus on alternating offers, reflecting protocols 
more commonly used in practice, to make progress on the problem
of a buyer, negotiating privately and bilaterally with multiple sellers, whilst taking interdependencies between the deals into account. These types of concurrent negotiations have been described as “one of the most important challenges for automated negotiation” [14].

There are several challenges that arise in such a setting:

1. In order to achieve beneficial deals with different sellers at the same time, the negotiation protocol must support concurrent interactions, allowing individual pairwise negotiations to evolve over time without blocking each other and supporting information exchange between them.
2. At the same time, a buyer should be able to align the efforts between parallel, interdependent negotiation threads and evaluate their aggregate result.
3. Since several sellers might offer the same product at different quantities or prices (as illustrated in Fig. 1), the buyer needs to decide which Sellers to negotiate with, and over what. Moreover, while doing so, the buyer needs to deal with complexities related to the combinatorial explosion of possible sets of partial deals.

Game theoretic techniques such as mechanism design and auctions have been used to tackle such challenges and can offer various guarantees on the outcome or optimal strategies. However, they fail to support less structured and more flexible settings often found in procurement [24], featuring two-way offers and counter-offers and different negotiation strategies per opponent. The problem has also been studied in iterative combinatorial auctions [23], however practical uptake may have been impeded by the fact that determining the revenue maximizing outcome for combinatorial bids is NP-complete [26]. Important work tackling procurement challenges has also been performed in the area of agent-based concurrent and/or one-to-many negotiations, albeit predominantly in a setting where one deal is sought (see also the next section for an overview). For instance, Rahwan et al. [25] were the first to introduce the coordination paradigm for single-deal one-to-many negotiations. Nguyen et al. improved this framework further by introducing status messages between coordinator and subnegotiators [19], and later improvements allow commitments [20], opponent information, and strategy tweaks from utility predictions [30].

To study multi-deal negotiations in a one-to-many setting in a principled way, we propose the Bargaining Chips framework: a general, multi-issue negotiation environment for handling offers and combining them, together with a protocol that governs the bilateral interactions. Bargaining Chips models a procurement setting by tasking a buyer agent with negotiating one-on-one with several sellers at once to obtain a number of chips of varying colors and prices. The buyer is able to obtain multiple deals through each of the concurrent bilateral interactions, with the goal of satisfying, in aggregate, a predefined demand for the lowest possible price.

Due to its modular design, Bargaining Chips can subsume various challenging settings proposed by other authors. For instance, when only one chip is being procured, the setting simplifies to the single-deal coordination problem first proposed by Rahwan et al. [25] and later extended by Nguyen et al. [20]. Furthermore, the bilateral interaction model is backward compatible with the alternating offers protocol and thus existing state-of-the-art agents (e.g. from the negotiation competition ANAC [5]) can be integrated in the Bargaining Chips framework, providing a flying start.

As a result, finding efficient negotiation and coordination strategies for the general setting of Bargaining Chips would address many challenges currently important in the research community.

In this paper, we will not propose solutions to all these challenges; however, we present classes of new, general-purpose multi-deal coordination strategies that showcase the architecture’s ability for tackling one-to-many negotiation challenges. We indicate new avenues for future work and show through simulation experiments that currently existing single-deal coordination strategies are outperformed by a set of new coordinators that are able to combine multiple deals.

2 RELATED WORK

Researchers have paid much attention to the coordination of concurrent one-to-many negotiations since Rahwan et al. highlighted that this approach offers increases scalability, reusability, and robustness [25]. We briefly cover the most relevant related work in this section and provide a comparison matrix for a selection of work in Table 1.

Nguyen and Jennings present a concurrent bilateral negotiation model involving a coordinator together with negotiation threads [19]. The coordinator determines the negotiation strategy for each thread based on heuristics that require prior knowledge about the environment and opponents. William et al. advance this work for unknown opponents and introduce a novel strategy that exploits the prediction of utility received in the future [30]. In these works, a buyer negotiates on the same set of multiple issues with different sellers and aims to get the single best deal for itself, while our framework enables composite (i.e., multi-deal) and asynchronous concurrent negotiations in which the buyer does not have to wait for all sellers to make decisions.

Similar to our work, Alrayes et al. present an asynchronous open market negotiation framework [1, 7]. However, they support only single-issue negotiations in which buyer agents can also reserve offers (with a penalty for canceling them later), while buyers can negotiate on multiple issues such as price and quantity in our framework. An et al. [2] use the relative scarcity of goods to steer the negotiation deadline of subnegotiators. In their model, multiple deals can be realized, but subnegotiators negotiate over a single issue. In another work of An et al. [3] their buyer, knowing the maximum quantities that each seller can offer, formulates its offers over price and the possible decommitment penalty. Their protocol uses double acceptance, however, it does not allow for asynchrony. Sim et al. [28] focus on interrelated markets in which a customer needs to combine a set of services from different providers; therefore, the buyer negotiates with them concurrently to utilize the desired composite service. Unlike our approach, they only negotiate over the price, while in our case, the buyer needs to negotiate over a bundle consisting of different quantities of chips, making the process more complex.

Mohammad et al. [17] use a negotiation platform for supply chain management where a factory manager agent acts as a buyer and negotiates concurrently with multiple suppliers bilaterally. This platform has recently focused on global utility estimation, and optimization techniques for concurrent multi-deal one-to-many
Table 1: Comparison matrix for selected concurrent one-to-many negotiation studies.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Single/Multi-deal</td>
<td>Multi-deal</td>
<td>Multi-deal</td>
<td>Multi-deal</td>
<td>Multi-deal</td>
<td>Single Deal</td>
<td>Multi-deal</td>
<td>Single-deal</td>
</tr>
<tr>
<td>Asynchronous</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Deadline per thread</td>
<td>Global</td>
<td>Global</td>
<td>Global</td>
<td>Global</td>
<td>Private</td>
<td>Global</td>
<td>Global</td>
</tr>
<tr>
<td>Action Order</td>
<td>Asynchronous</td>
<td>Turn-taking</td>
<td>Turn-taking</td>
<td>Turn-taking</td>
<td>Two side</td>
<td>Turn-taking</td>
<td>Turn-taking</td>
</tr>
<tr>
<td>Acceptance</td>
<td>Two side</td>
<td>One side</td>
<td>One side</td>
<td>One side</td>
<td>Two side</td>
<td>One side</td>
<td>One side</td>
</tr>
<tr>
<td>Decommitment</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Main Focus</td>
<td>Identifying coordination mechanisms</td>
<td>Global utility estimation &amp; optimization</td>
<td>Formulating the QoE problem as one-to-many negotiation</td>
<td>Negotiation procedures for interdependent negotiation</td>
<td>Proposing a negotiation strategy for e-markets</td>
<td>Comparing coordination strategies in resource markets</td>
<td>Proposing a coordination mechanism &amp; strategy</td>
</tr>
</tbody>
</table>

Negotiations [16]. For the bilateral negotiations, Mohammad et al. adopt the Alternating Offer Protocol [22], where agents negotiate in a turn-taking fashion. Our approach differs in that agents do not need to wait for each other’s actions (i.e., they send out offers at any time asynchronously). In that sense, our work is similar to [13] where De Jonge and Sierra present an unstructured multilateral negotiation protocol enabling agents to specify freely what offer(s) are acceptable in an asynchronous way and to keep multiple offers on the negotiation table. In our work, an agent’s offer overrides the previous ones.

Another study addressing the multiple interdependent negotiations is proposed by Niu, Ren and Zhang [21]. They present three different procedures: sequential, concurrent, and clustered negotiations, where the Alternating Offers protocol is adopted within each bilateral interaction. In our case, agents can take valid actions at any time without waiting for their opponent’s response. Najar et al. similarly adopt the alternating offers protocol in their formulation of the Quality of Evaluation problem as one-to-many negotiations and define utility functions and strategies accordingly [18]. Furthermore, Divekar et al. introduce a dialogue-based negotiation framework in which a human buyer negotiates with multiple seller agents [16]. The buyer can make multiple deals with different sellers concurrently, but the agents can take at most one action per round, and all offers are made public—which might not be convenient for e-markets.

It is worth mentioning that The Colored Trails (CT) game is another well-known human-agent framework designed for analyzing decision-making strategies of agents on exchanging their resources [11]. Agents exchange brightly colored chips to reach their goals, but in contrast to our work, each game consists of a one-shot bilateral negotiation without any counter-offers.

Lastly, in this work rather than proposing a concrete coordination strategy, as previously in literature, we identify some components that each coordination strategy in a concurrent setting must have.

### 3 Negotiation Model

The Bargaining Chips negotiation model introduces a setting in which a buyer attempts to satisfy their demand through concurrent bilateral negotiations with multiple sellers, exchanging offers through an asynchronous negotiation protocol that prescribes what offers are valid at each stage of the negotiation.

#### 3.1 Domain Model

**Bargaining Chips** is played by a buyer who seeks to acquire a number of chips for a good price. Chips represent arbitrary indivisible items, such as products or tasks and are differentiated from each other by a unique color, which is a discrete value $c \in C$.

Different sellers supply chips in bundles of varying quantity and unit price; for example, a seller may have 2 red chips for $3 each and 1 blue chip for $4 on offer (which we denote as $2 \times \color[RGB]{204,0,0} \text{red}, 1 \times \color[RGB]{0,0,255} \text{blue}$). Formally, a bundle $b$ is a triple $(B, p, q)$, where $B \subseteq C$, $p : B \rightarrow \mathbb{R}_{\geq 0}$ assigns the unit price to each chip in $B$, and $q : B \rightarrow \mathbb{N}$ prescribes each chip’s quantity. Note that bundles can be extended to general multi-issue items [12, 15] by extending the definition with arbitrary additional attributes such as size, weight, expiration date, and so on.

The bundle’s (total) price is the grand total of all prices; i.e., $\text{total price}(b) = \sum_{c \in B} q(c)p(c)$. Note that while exchanging information about the bundle’s total price alone may suffice in principle, listing the unit price of each individual item exposes more information to the buyer and seller, thereby aiding the negotiation process. If needed, additional compound properties can be expressed similarly, such as shipment cost, delivery time, bulk discount, total weight, and so on.

Together, the set $\Omega$ of all bundles considered by the agents is called the outcome space or the possible agreements. We always consider the empty bundle $\emptyset$ as a special kind of possible outcome. The buyer possesses a utility function $u : \Omega \rightarrow [0, 1]$ over the outcome space. One of the important factors determining the buyer’s utility is their demand specifying the desired number for every chip; for example, the buyer might be after 2 red chips and 1 blue chip (denoted $\color[RGB]{204,0,0} \text{red}, \color[RGB]{0,0,255} \text{blue}$).

#### 3.2 The negotiation protocol

To reach an agreement, the buyer and sellers exchange offers with each other according to the rules specified by the negotiation protocol. An important desideratum of the negotiation protocol is that the buyer is able to interact freely with all sellers at the same time without getting blocked by any of the concurrent interactions. Currently, the most used model for making offers in single-deal negotiations uses the alternating multi-issue offers protocol [27], where a negotiation is a back-and-forth offering of values in $V_1 \times \cdots \times V_m$.
of negotiable issues $1, \ldots, m$. However, this protocol and recent extensions [6] cannot be used readily in the concurrent setting of procurement, because of three problems: 1) the alternating offers protocol requires the negotiators to wait, possibly indefinitely, for the opponent to respond. This means that it is impossible to renew an offer, even when required by circumstances in other ongoing interactions; 2) there is no mechanism in place to strike and combine multiple agreements in a general way; 3) each offer is binding and can therefore be accepted by the other side at any moment, which impedes the possibility of putting out offers that are mutually exclusive.

To tackle these challenges, we propose a new negotiation protocol called the Asynchronous Offers Protocol to handle each bilateral interaction. The Asynchronous Offers Protocol achieves a concurrent, live interaction by allowing each side to send out an offer asynchronously (and possibly multiple times in a row, overriding the previous bid), combined with a double-accept mechanism that concludes in an agreement when both sides accept the final proposal. In the Asynchronous Offers Protocol, actions can occur more often, so that each interaction can continue without having to wait for a response or progress in any of the other negotiations.

Formally, let two agents $\mathcal{A} = \{b, s\}$ be given, where $b$ is the buyer and $s$ is a seller. For an agent $a \in \mathcal{A}$, we will denote by $\bar{a} \in \mathcal{A} \setminus \{a\}$ its counterpart. The offers $o \in O$ exchanged between $a$ and $\bar{a}$ can be bundle, or a special message that signals acceptance, acknowledgment of acceptance, or acceptance, or walking away from the negotiation:

$$O = \Omega \cup \{\text{Accept}(o) : o \in \Omega\} \cup \{\text{Ack}(o) : o \in \Omega\} \cup \{\text{End}\}.$$  
An offer is always directed, and when an offer $o \in O$ is sent by agent $a \in \mathcal{A}$ to its counterpart, we represent this as $\sigma^a \rightarrow \bar{a}$ when we want to make this explicit. An ordered sequence of offers

$$T = (\sigma^a_1 \rightarrow \bar{a}, \ldots, \sigma^a_t \rightarrow \bar{a})$$

between the agents is called a negotiation thread.

**Definition 3.1 (Outcome of a negotiation thread).** Given a negotiation thread $T = (\sigma^a_1 \rightarrow \bar{a}, \ldots, \sigma^a_t \rightarrow \bar{a})$, we call the thread $T$ failed when $o_t = \text{End}$ or when $t \geq D$ has exceeded a predefined deadline $D$. The outcome $O(T)$ of a failed thread $T$ is defined as the empty bundle $\emptyset$. $T$ is successful when $t \leq D$ and $o_t = \text{Ack}(b)$, for a bundle $b \in \Omega$. This bundle $b$ is called the agreement and defines the outcome $O(T) = b$ of a successful thread $T$. In both cases, the thread is said to be concluded. When $o_t = \text{Accept}(b)$ we call the thread pending. All other threads are ongoing with outcome $O(T) = \emptyset$.

The negotiating agents at both sides can send offers to each other while the thread is ongoing and they can send multiple offers in a row. A deal is binding when agent $a$ accepts a bundle offered by $\bar{a}$, and the accept is in turn acknowledged by $\bar{a}$. However, not every sequence of actions is allowed; negotiation threads are valid under conditions defined below.

**Definition 3.2 (The Asynchronous Offers Protocol).** Let a thread $T = (\sigma^a_1 \rightarrow \bar{a}, \ldots, \sigma^a_t \rightarrow \bar{a})$, with $t \leq D$ be given. Let $o$ be an arbitrary offer in the thread sent by $a$; i.e. choose any $i \in \{1, \ldots, t\}$ such that $a_i = a$ and set $o = \sigma^a_i \rightarrow \bar{a}$. Let $\bar{b}$ be the offer that was last sent by $\bar{a}$ before $o$; i.e. let $j \in \{1, \ldots, i - 1\}$ be the highest index such that $a_j = \bar{a}$ and set $\bar{b} = \sigma^{\bar{a}}_j \rightarrow a$ (and in case there is no such previous offer, we set $\bar{b} = \emptyset$). The thread $T$ is valid if it holds that:

1. **Bid:** If $o = \emptyset$, then $\bar{b} \in \Omega \cup \{\text{Accept}(b) : b \in \Omega\}$.
2. **Accepting bid:** If $o = \text{Accept}(b)$, then $b = \bar{b} \in \Omega \setminus \{\emptyset\}$.
3. **Acknowledging accept:** If $o = \text{Ack}(b)$, then $i = t$ and $\bar{b} = \text{Accept}(b)$.
4. **End:** If $o = \text{End}$, then $i = t$ and $\bar{b} \in \Omega \cup \{\text{Accept}(b)\}$.

Note how the Asynchronous Offers Protocol is rather flexible: actions can occur at any time, in any order, without waiting for each other, and offers and even accepts can effectively be retracted by proposing a new offer that overwrites the previous one. This makes it possible to signal interest in an offer without committing to it, allowing for an adaptive interaction dynamic where negotiators can respond quickly to changes in the market (i.e. belief change induced by new developments in other ongoing threads). Note that the Asynchronous Offers Protocol defines a proper extension of the Alternating Offers Protocol and avoids livelock as well as the need for special de-commitment actions used in many concurrent negotiation approaches where a one-sided accept is considered binding [8, 20, 29]. In game theoretic terms, the double-accept mechanism makes preceding offers cheap talk, which can extend the set of potential equilibria.

## 4 COORDINATING MULTIPLE DEALS

Bargaining Chips fuses multiple bilateral negotiations into a concurrent, one-to-many negotiation setting by viewing it as a coordination problem, in which the buyer needs to aggregate and coordinate multiple, overarching agreements such that the composite outcome satisfies the buyer’s overall demand.

### 4.1 Aggregating offers

The buyer procures bundles from each seller individually with the goal of aggregating several bilateral agreements. We define the aggregation operator for two bundles $b_1 = (B_1, p_1, q_1)$ and $b_2 = (B_2, p_2, q_2)$ as follows:

$$b_1 \oplus b_2 := (B_1 \cup B_2, p, q),$$

so that all chips are added:

$$\text{q}(c) := \begin{cases} q_1(c), & c \in B_1 \cap B_2 \\ q_2(c), & c \in B_1 \setminus B_2 \\ \text{otherwise}, & \text{otherwise}, \end{cases}$$

and the aggregate unit price is the updated weighted average:

$$\text{p}(c) := \begin{cases} p_1(c)q_1(c) + p_2(c)q_2(c), & c \in B_1 \cap B_2 \\ p_1(c), & c \in B_1 \setminus B_2 \\ p_2(c), & \text{otherwise}. \end{cases}$$

**Example 4.1 (Bundle aggregation).** Assume a deal is reached with two sellers: $b_1 = (3 \times 5, 3 \times 5, 3 \times 5)$ and $b_2 = (1 \times 5, 7 \times 5, 7 \times 5)$. The aggregation of these two bundles results in $b_1 \oplus b_2 = (4 \times 5, 3 \times 5, 5 \times 5)$.

Since $\oplus$ is commutative and associative we may safely string together a set of bundles $B$ with the inductively defined $\bigoplus_{b \in B}$
4.2 Combining multiple deals

To be able to strike multiple deals, a buyer $b$ needs to engage in multiple negotiations with sellers $s_1, \ldots, s_m$ and combine their outcome. To do so (see Fig. 2), the responsibility for making offers is delegated to subnegotiators $b_1, \ldots, b_m$, such that the one-to-many negotiation consists of $m$ concurrent bilateral negotiation threads $T_i$ between $b_i$ and $s_i$. The coordinator aligns the actions of the subnegotiators between the threads and combines their deals to ensure a good aggregate outcome.

A one-to-many negotiation $N = \{T_1, \ldots, T_m\}$ thus consists of $m$ concurrent bilateral negotiation threads $T_i$ between $b$ and $s_i$. The outcome $O(N)$ of the negotiation $N$ is computed by aggregating the negotiation outcomes $O(T_1), \ldots, O(T_m)$ in each thread:

$$O(N) = \bigoplus_{i=1}^{m} O(T_i).$$

The buyer’s goal is to maximize the utility $u(O(N))$ of the aggregate outcome of the entire negotiation. Note that the interdependencies between the threads are codified by the specifics of the aggregation operator. Failing to consider these interdependencies could lead to paying an unnecessary premium, or realizing undesirable quantities, deteriorating utility. Since ongoing threads produce an empty outcome, the aggregated outcome can be calculated at any time to represent what has been obtained by the buyer so far.

Each subnegotiator $b_i$ is a typical bilateral negotiation agent (e.g. taken from the ANAC repository [5]) and is designated its own objectives through its utility function $u_i$ (typically initialized by the coordinator to $u_i = u$). This modular design provides several advantages compared to a monolithic agent design, including increased simplicity, reusability, and robustness [25]. Furthermore, this approach mirrors real-life procurement settings, where managers typically divide tasks over multiple human negotiators. The most important difference between our design and the architecture of [25] is that our model allows for composite negotiations, in which the outcome can comprise multiple partial outcomes, each originating from a deal with a different seller.

4.3 Coordination and status messages

The coordinator ensures a desirable, composite deal by exchanging messages regularly with each subnegotiator. As depicted in Fig. 2, the coordinator can send coordination messages that contain directives for realizing the appropriate part of the buyer’s demand; whereas the coordinator receives status messages containing updates regarding the status of each negotiation thread.

Typical coordination messages signal actions to be performed by the subnegotiators: e.g., halting the negotiation, granting permission to accept certain outcomes, updating the subnegotiator’s goals, or updating the subnegotiator’s strategy parameters. Possible status messages include: reaching an outcome, pending an agreement, predicting an outcome, or timer-based events. Each type of message defines a set of percepts and actuators of the coordinator agent, thereby defining a different decision-making problem.

4.4 Coordination strategies for pending agreements

The Bargaining Chips model can in principle support any type of status and coordination messages. However, in this work, we propose and classify a set of coordinators that can act on a general set of messages about an agreement being reached. That is, coordinators receive a status message when a subnegotiator wishes to send out an offer that moves the thread into a concluded or pending state. Recall from Definition 3.1 that the negotiation threads in $N$ can be split up according to three distinct states: the concluded ($C$), the pending ($P$), and the ongoing ($G$) threads, with $N = C \cup P \cup G$. When a coordinator learns through status messaging that potential agreements are available, coordination messages can be sent in response to direct a subnegotiator to accept the pending offer, update its utility function and continue, or end the negotiation. We believe that agreement-based coordination strikes a good balance in messaging frequency and expressive power, since these events are essential decision points in the negotiation that do not occur very often.

We propose three traits that describe distinct parts of such multi-acceptance coordination strategies:

- **Timing (Patient – Desperate).** A coordinator can choose when to make a decision: i.e., whether to allow or reject a pending agreement right now, or to postpone the decision until later (e.g. when more agreements are available).

  At one end of the spectrum is the Desperate strategy, which responds immediately with either an Accept or End. At the other extreme, the Patient strategy waits until all negotiation threads are concluded or pending and then makes a decision. In principle of course, other timing strategies are also possible, such as postponing action until a certain time interval, or waiting for agreements to be pending for all sellers of a
particular item. This generalizes Rahwan’s classification for single deals [25].

- **Selecting (Greedy – Look-ahead).** For each thread that is pending, the coordinator needs to make a decision for the (possibly singleton) set of potential agreements. An obvious choice is using a Greedy strategy, in which the pending threads \( T \in P \) are approved iteratively by the coordinator while their marginal utility contributes positively:

\[
u(O(C \cup T)) > u(O(C)).
\]

When it does not, the pending thread is sent a termination message instead. Alternatively, the Look-ahead strategy maximizes pending threads all at once, selecting the subset with the highest aggregate utility given what has been procured already:

\[
\max_{A \subseteq P} u(O(A \cup C)).
\]

- **Control (Open-loop – Closed-loop).**

When worthwhile deals have been selected, the coordinator may exert additional control over the other threads by updating them with information that becomes available from the other threads. We distinguish an Open-loop coordinator that does not send out any updates, and the Closed-loop coordinator that does. An important example is updating the utility function \( u_i \) of each subnegotiator \( b_j \) by incorporating the deals that have already been struck elsewhere:

\[
u_i(x) := u(x @ O(C)).
\]

We illustrate the above classification by detailing the decision-making process of four coordinators composed of the above types (see Table 2), where all lead to a different composite deal.

**Example 4.2 (Coordinator types).** Imagine a simplified scenario in which the buyer negotiates with 4 sellers to obtain 10 chips of the same color, regardless of their price. Suppose the first subnegotiator manages to procure 6 chips, and after that, the second subnegotiator procures 8 chips, and so on, creating the following sequence of pending agreements: \( (6, 8, 5, 5) \). The Desperate coordinator will evaluate the deals one-by-one, obtaining 11 chips in an open loop (listed under \( O(N) \) in Table 2), while the Patient coordinators obtain 8 and 10 chips by evaluating all deals at once, but in different ways. For the Closed-loop coordinator, the sequence of pending agreements will look different, since the utility functions are updated each time a new bilateral agreement is reached. For example, after accepting the first offer of 6, the goals of the other three subnegotiators is updated to collect 4 more chips. As a result, the second subnegotiator might secure 3 instead of 8, and, continuing the closed-loop cycle in this way, procure 9 chips overall.

It is not immediately obvious which combination of traits is beneficial in complex coordinated one-to-many negotiations since trade-offs are involved in each of them. In general, making a later and more informed decision is beneficial, but holding out for too long may lead to sellers dropping out, or worse, to a failure in securing all available deals in time. Similarly, subnegotiators stand to gain from up-to-date information, but often at the cost of having to recalculate the internal negotiation model and/or restart the negotiation from scratch. Note also that not all combinations of the above traits are meaningful; for instance, a Desperate Coordinator cannot select deals using Look-ahead selection, since there is only one decision to be made at any given time step. In addition, market circumstances can have a big impact on the performance of specific coordinators. For instance, when suppliers sell in bulk it is more difficult to reach small partial deals, which diminishes the advantages of closed-loop coordination. We investigate these trade-offs in detail in our experiments below.

## 5 EXPERIMENTS

To compare our coordinator strategies and evaluate their performance in a simulation, we conduct two experiments using a Java implementation of the Bargaining Chips framework. In the first experiment, we examine how different coordinators perform under idealized conditions in which a coordinator can accept an offer at any time before the negotiation deadline. In the second experiment we introduce the risk of offers expiring to investigate the robustness of different coordinators.

### 5.1 Setup

We compare the four multi-deal coordinators of example 4.2, together with two baseline single-deal coordinators proposed by Rahwan [25] (Desperate Single-deal and Patient Single-deal), in a scenario that involves a buyer seeking 10 different types of chips from 10 sellers. We evaluate the performance of each coordinator over the same set of configurations in over 20000 simulations, measuring average normalized buyer’s utility. To normalize utility, we estimate an upper bound by solving the scenario as a centralized constrained optimization problem, where we find the optimal set of bundles that are both acceptable to each seller and maximizes the buyer’s utility.

We assign to the buyer a goal bundle \( g \) composed of all chips, with quantities \( q \in \{10, \ldots, 15\} \) each, and a constant unit goal price \( (p = 1) \). The preferences of the coordinator are expressed through an additive utility function \( u \) constructed around \( g \) as follows: for each chip \( c \), we construct triangular utilities around the goal quantity and price, specifying their end points by multiplying each peak value with the corresponding amount over which the buyer is willing to deviate from its goal, called the tolerance parameter \( \delta_q \) (lower quantity tolerance \( \delta_q^l \in [0.5, 1] \), upper quantity tolerance \( \delta_q^u \in [0, 0.5] \), lower price tolerance \( \delta_p^l = 0 \), and upper price tolerance \( \delta_p^u = 1 \)). We define the utility \( u_c \) over a single chip \( c \) as:

\[
u_c(q(c), p(c)) = w_c^q u_q^c(p(c)) + w_c^p u_p^c(q(c))
\]

### Table 2: The decisions \( d_j \) of each coordinator type, resulting in the composite negotiation outcome \( O(N) \). Green numbers stand for accepted offers and red for the refused ones.

<table>
<thead>
<tr>
<th>Coordinator</th>
<th>( d_1 )</th>
<th>( d_2 )</th>
<th>( d_3 )</th>
<th>( d_4 )</th>
<th>( O(N) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desperate Greedy Open-loop</td>
<td>6</td>
<td>8</td>
<td>5</td>
<td>5</td>
<td>11</td>
</tr>
<tr>
<td>Patient Greedy Open-loop</td>
<td>6, 8, 5, 5</td>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Patient Look-ahead Open-loop</td>
<td>6, 8, 5, 5</td>
<td></td>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Desperate Greedy Closed-loop</td>
<td>6</td>
<td>3</td>
<td>4</td>
<td></td>
<td>9</td>
</tr>
</tbody>
</table>
where \( w^b \in (0, 1) \) and \( w^q + w^b = 1 \). Finally we define the utility of the coordinator as:

\[
\sum_{c \in C} w_c \cdot u_c(p(c), q(c)), \quad b \in B,
\]

where \( C \) is the set of chips in bundle \( b \), \( w_c \in (0, 1) \) are randomly chosen weights such that \( \sum_{c \in C} w_c = 1 \), and \( B \) is the set of bundles the coordinator is willing to consider, constructed using \( g \) and the tolerance parameters.

The buyer’s goal quantity is multiplied by \( 0.5 \), divided among the sellers, making sure that the percentage of sellers that can offer each chip \( \%_{s/c} \in [10\%, 100\%] \), while also perturbing their prices (sellers goal price \( p_s \in \{2, 3\}) \). We use this to generate a sequence of 100 random sub-bundles per seller, which are offered iteratively to the buyer.

For the sub-negotiators we constrain the coordinator’s goal bundle to the colors that their corresponding seller can provide and follow the same procedure to construct their utility functions. Each sub-negotiator accepts a bid that brings positive utility.

The combination of the bilateral strategies of the sellers and sub-negotiators guarantees that in every scenario, all open-loop coordinators receive the same sequence of offers. The offers received by the closed-loop coordinator can differ, as each time the closed-loop coordinator approves a bilateral agreement, it updates the utility function \( u_b \) of each sub-negotiator \( b \) by setting its peak to the updated goal bundle.

### 5.2 Results

#### 5.2.1 Experiment 1 - Permanent Offers

We start with a setting where deals procured by sub-negotiators do not expire, while we vary the availability of each chip \( \%_{s/c} \). Our results show a clear ranking among some of the coordination traits (see Fig. 3). When open-loop control is used, the patient multi-deal coordinators are in the lead (Wilcoxon signed-test\(^2\), \( p\)-value < 0.01), since they benefit from taking a decision to accept only after all their sub-negotiators have an available deal each. Furthermore, look-ahead selection identifies the set of combined bilateral deals that maximize the buyer’s utility and therefore achieve better performance (\( p\)-value < 0.01) compared to greedy selection when there are enough sellers that offer the same chips (at least 20% of the sellers). Lastly, the single-deal baseline strategies from Rahwan et al. trail behind significantly (\( p\)-value < 0.01), as the multi-deal coordinators make use of 5.8 bilateral deals on average, which increases their aggregate utility considerably.

For most of the values of \( \%_{s/c} \), there is no significant difference in performance between the patient look-ahead open-loop coordinator and its polar opposite, the desperate greedy closed-loop coordinator (i.e. insignificant for the interval \([0.4, 0.8]\) and small in \([0.8, 1.0]\)). However, this may change in more realistic circumstances where offers might expire if the coordinator takes too long (for instance because sellers might reach deals with other buyers).

#### 5.2.2 Experiment 2 - Expiring Offers

As a second experiment, we explore a setting where there is a probability \( p_e \) that some offers

---

\(^2\) We tested the significance of the results using nonparametric Wilcoxon signed-test for each pair of coordinators, because Shapiro-Wilk tests showed the data is not normally distributed.

---

**Figure 3:** Average renormalized utility as we vary the percentage of sellers that can offer each chip, over 1000 iterations each. The colored bounds display the confidence intervals.

**Figure 4:** Average renormalized utility as we vary offer expiration probability \( p_e \) over 1000 iterations.
We are indebted to Flavio Gaier and Niklas Hall from Acumex ApS. This work introduces Bargaining Chips: a concurrent one-to-many negotiation framework for studying multiple composite bilateral deals in a procurement setting. We present a new model for aggregating deals and design a new protocol for exchanging offers asynchronously. We also propose and compare a set of general coordination strategies for our setting, showing that the ability to secure multiple deals results in better overall outcomes, and that desperate coordination combined with a closed-loop control outperforms other coordination-traits combinations, especially in markets where offers can expire.

Future work may expand on our initial exploration of this complex setting by accounting for other facets of real-world procurement, such as compound bundle attributes (e.g. shipping costs). Coordination strategies could be further improved by using predictions (e.g. by integrating work similar to that of [30]) so that coordinators can act with additional information before agreements are reached.

Furthermore, in this work we consider rather straightforward base negotiation behaviors for the bilateral interactions, such as concession strategies, for which one may consider alternatives that strategize more. The embedding of Bargaining Chips in external factors can be refined by accounting for (1) a model market, estimating how active competing buyers are behaving; and (2) a user model, estimating the preferences of the user the agent represents.

Finally, we are hopeful the Bargaining Chips framework together with our class of benchmark multi-deal coordinators can help progress towards solving the multi-protocol challenges more generally, thereby broadening the real-world applicability of automated negotiation.

ACKNOWLEDGMENTS

We are indebted to Flavio Gaier and Niklas Hall from Acumex ApS for their help in devising a realistic procurement model and for the inspiration for the Asynchronous Offers Protocol. The research reported in this article is part of Vidi research project VI.Vidi.203.044, which is financed by the Dutch Research Council (NWO).

REFERENCES
