Chapter 2

Expressive Preference Aggregation Settings

Freedom is, first of all, the chance to formulate the available choices, to argue over them – and then, the opportunity to choose.

C. Wright Mills

When the preferences of multiple agents need to be aggregated to choose an outcome (such as an allocation of resources), we require some process for doing so. One option is *ad hoc* negotiation, where the agents controlling the relevant resources attempt to improve the outcome locally, by proposing and accepting deals to each other as they see fit. The upsides of this approach are that no centralized computation is needed, and that the lack of constraints associated with this type of negotiation potentially allows the agents some amount of creativity in adapting to circumstances. The downside is the lack of guidance and oversight that the agents are confronted with. When deciding what deal to propose next (or whether to accept another agent's proposal), an agent needs to assess what is likely to happen in future negotiation. However, the unstructured approach of *ad hoc* negotiation makes this exceedingly difficult. In addition, the contract language needs to be complex to avoid getting stuck in local optima, making the process even less overseeable. As a result, in all but the simplest of settings, agents will likely only make ineffective deals and not come anywhere close to reaching the optimal outcome. (Early work by Sandholm and others [Sandholm, 1993b; Sandholm and Lesser, 1995; Sandholm, 1997; Andersson and Sandholm, 1999, 2000; Sandholm and Lesser, 2002] provides a more formalized version of such a distributed process.)

Another approach is to have a clear *protocol* that elicits information about the agents' preferences in a predictable, transparent, and commonly known manner to arrive at an outcome. Well-known examples include auction protocols (such as a *first-price sealed-bid auction*, where every agent submits a bid for the good for sale in a sealed envelope, and the highest bidder wins the item for the price she specifies), as well as voting protocols (such as the *Plurality* protocol, where everyone submits her most preferred candidate, and the candidate with the most votes wins). The clarity, transparency, and predictability of such protocols make it possible for agents to assess the likelihoods of future events and act in accordance. Unfortunately, naïvely designed protocols run the risk of being overly restrictive in the negotiation that they allow. For instance, suppose there are

two items for sale, and we auction them off individually and sequentially. One bidder may consider the items complementary: neither item by itself would be useful to her, but together they are worth something. This bidder may be hesitant to bid high in the first auction, for fear that another bidder will win the second item—leaving her stuck with only the first item. This hesitancy may prevent her from winning the first item, even if the economically efficient outcome is for her to win both items. A likely event in this scenario is that the bidder seeks to strike a deal with the seller to buy both items outside of the auction, thereby reverting to *ad hoc* negotiation and the problems it entails.

The solution, of course, is to make sure that the protocols are not deemed too restrictive by the agents. In the example, the two items could be auctioned off simultaneously in a combinatorial auction, allowing bids on the bundle of both items. Protocols such as combinatorial auctions that allow the agents to express their full preferences, and that act on that information, are known as *expressive preference aggregation* protocols. In recent years, billions of dollars have been saved by applying such protocols to strategic sourcing [Sandholm *et al.*, 2006; Sandholm, 2006].

This dissertation will not consider any *ad-hoc* approaches to preference aggregation. Rather, it will focus on clear protocols that allow the agents to provide their preference information in expressive languages. The remainder of this chapter will introduce some preference aggregation settings, together with corresponding languages in which agents can express their preferences and criteria according to which the outcome can be selected. We will discuss computational aspects of these settings in later chapters. For example, Chapter 3 will discuss the computational complexity of and algorithms for choosing the optimal outcome in these settings. Some of the results in later chapters will not be specific to any particular setting, but the settings introduced in this chapter can serve as example domains.

The rest of this chapter is layed out as follows. In Section 2.1, we discuss *voting* (or *rank aggregation*) as an approach to preference aggregation. Here, each agent simply ranks all possible outcomes, and the outcome is chosen based on these rankings according to some *voting rule* (some example voting rules will be given). In Section 2.2, we discuss *allocation of tasks and resources*, and the use of combinatorial auctions and exchanges for doing so. In Section 2.3, we introduce a new application: letting multiple potential donors negotiate over who gives how much to which of multiple (say, charitable) causes [Conitzer and Sandholm, 2004e]. Finally, in Section 2.4, we study preference aggregation in settings with externalities and introduce a representation, a language for expressing agent preferences, and criteria for choosing an optimal outcome [Conitzer and Sandholm, 2005d].

2.1 Voting over alternatives (rank aggregation)

A very general approach to aggregating agents' preferences over outcomes is the following: let each agent rank all of the alternatives, and choose the winning alternative based on these rankings. (In some settings, rather than merely producing a winning alternative, one may wish to produce an aggregate ranking of all the alternatives.) This approach is often referred to as *voting* over the alternatives, and hence, in this context, agents are referred to as *voters*, the rankings that they submit as *votes*, and the alternatives as *candidates*.

For example, in a setting with three candidates a, b, c, voter 1 may vote $a \succ b \succ c$, voter 2 $b \succ a \succ c$, and voter 3 $a \succ c \succ b$. The winner (or aggregate ranking of the candidates) depends on

which *voting rule* is used. Formally, letting C be the set of candidates, $R(C)$ the set of all possible rankings of the candidates, and n the number of voters, a voting rule is a mapping from $R(C)^n$ to C (if one only wishes to produce a winner) or to $R(C)$ (if one wishes to produce an aggregate ranking). One example rule is the *plurality* rule, where candidates are ranked simply according to how often they are ranked first by voters. In the example, a is ranked first twice, b once, and c never, so that the aggregate ranking produced by the plurality rule is $a \succ b \succ c$. Under the plurality rule, the voters effectively vote only for a single candidate (how the voter ranks the candidates below the top candidate is irrelevant).

Rules such as plurality may leave candidates tied, and typically these ties will need to be broken somehow (especially to choose a winning alternative). Throughout, we will make as few assumptions as possible on how ties are broken, but where we do make assumptions, we will make this clear. One may also wonder if we can allow for candidates to be tied in the *votes*. It is typically not difficult to extend voting rules and results to allow for this, but we will assume throughout that rankings are total orders on the candidates, *i.e.* they have no ties. (Some recent work has addressed extending voting theory to settings in which voters submit *partial orders* [Pini *et al.*, 2005; Rossi *et al.*, 2006]; this is significantly more involved than merely allowing for ties.)

But why should one use the plurality rule? Perhaps it would be desirable to give a vote's second-ranked candidate some points, or even to use a rule that is not based on awarding points to the candidates at all. We will see examples of such rules shortly. First, however, let us consider if perhaps there exists an "ideal" rule. If there are only two candidates, it is clear what the voting rule should do: the candidate that is ranked higher more often should win. This leads us to the following idea: for any pair of candidates, we can see which one is ranked more often. For instance, in the above example, a is ranked above b twice, whereas b is ranked above a only once—hence we say that a wins the *pairwise election* between a and b. Similarly, a defeats c in their pairwise election, and b defeats c. Hence, naturally, the aggregate ranking should be $a \succ b \succ c$ (which agrees with the plurality rule).

However, this line of reasoning is not always sufficient to produce a ranking (or even a winner). Consider a modified example in which voter 1 votes $a \succ b \succ c$, voter 2 $b \succ c \succ a$, and voter 3 $c \succ a \succ b$. Now, a defeats b in their pairwise election, b defeats c, and c defeats a—that is, we have a cycle, and our aggregate ranking cannot be consistent with the outcomes of all pairwise elections. This is known as a *Condorcet paradox*, and it shows that, unfortunately, in deciding whether a should be ranked higher than b in the aggregate ranking, we cannot simply ignore the position of c in the rankings.

Indeed, a famous theorem by Arrow [1963] states that there is no deterministic voting rule (for producing an aggregate ranking) that has all of the following properties:

- The rule is *non-dictatorial*, that is, at least two voters have the potential to affect the outcome.
- The rule is consistent with *unanimity*, that is, if all voters prefer a to b, then the aggregate ranking must rank a above b as well.
- The rule satisfies *independence of irrelevant alternatives*, that is, which of two alternatives is ranked higher in the aggregate ranking should be independent of how the other alternatives are ranked in the votes.

Arrow's theorem and the possibility of Condorcet paradoxes depend on the voters' being unrestricted in how they order the candidates. One well-known restriction that makes these problems disappear is *single-peakedness* of the voters' preferences. We say that preferences are single-peaked if there is a total order $<$ on the candidates, and for any voter i and any three candidates $a < b < c$, $a \succ_i b \Rightarrow b \succ_i c$ and $c \succ_i b \Rightarrow b \succ_i a$. In words, the candidates are arranged on a spectrum from left to right, and a voter never prefers a candidate that is further from the voter's most preferred candidate (the voter's "peak") to a closer one. (Note that this definition does not compare candidates on the left side of a voter's peak with those on the right side in terms of closeness, that is, the notion of "closer to the peak" only applies to pairs of candidates that are on the same side of the peak. Also note that the order $<$ must be the *same* for all voters.) If the voters' preferences are single-peaked, then there are no Condorcet cycles. If we order the voters by their peaks, then the peak of the voter in the middle of this ordering (the *median* voter) will win all pairwise elections, that is, it is a *Condorcet winner*. (This is assuming that a median voter exists, *i.e.* the number of voters is odd.)

Nevertheless, in many settings the votes do not have any (apparent) structure, so that it is still important to define voting rules for the general case. Next, we review the most common voting rules. We will define them according to how they rank candidates; the winner is the top-ranked candidate.

- *Scoring rules*. Let $\vec{\alpha} = \langle \alpha_1, \ldots, \alpha_m \rangle$ be a vector of integers. For each vote, a candidate receives α_1 points if it is ranked first in the vote, α_2 if it is ranked second, *etc.* Candidates are ranked by their scores. The *Borda* rule is the scoring rule with $\vec{\alpha} = \langle m-1, m-2, \ldots, 0 \rangle$. The *plurality* rule is the scoring rule with $\vec{\alpha} = \langle 1, 0, \ldots, 0 \rangle$. The *veto* rule is the scoring rule with $\vec{\alpha} = \langle 1, 1, \ldots, 1, 0 \rangle$.

- *Single transferable vote (STV)*. This rule proceeds through a series of $m-1$ rounds. In each round, the candidate with the lowest plurality score (that is, the fewest votes ranking it first among the remaining candidates) is eliminated (and each of the votes for that candidate "transfers" to the next remaining candidate in the order given in that vote). The candidates are ranked in reverse order of elimination.

- *Plurality with run-off*. In this rule, a first round eliminates all candidates except the two with the highest plurality scores. Votes are transferred to these as in the STV rule, and a second round determines the winner from these two. Candidates are ranked according to Plurality scores, with the exception of the top two candidates whose relative ranking is determined according to the runoff.

- *Maximin* (aka. *Simpson*). For any two candidates a and b, let $N(a, b)$ be the number of votes that prefer a to b. The *maximin score* of a is $s(a) = \min_{b \neq a} N(a, b)$—that is, a's worst performance in a pairwise election. Candidates are ranked by their scores.

- *Copeland*. For any two candidates a and b, let $C(a, b) = 1$ if $N(a, b) > N(b, a)$, $C(a, b) = 1/2$ if $N(a, b) = N(b, a)$, and $C(a, b) = 0$ if $N(a, b) < N(b, a)$. The *Copeland score* of candidate a is $s(a) = \sum_{b \neq a} C(a, b)$. Candidates are ranked by their scores.

- *Bucklin*. For any candidate a and integer l, let $B(a, l)$ be the number of votes that rank candidate a among the top l candidates. For each candidate a, let $l(a)$ be the lowest l such that $B(a, l) > n/2$. Candidates are ranked inversely by $l(a)$. As a tiebreaker, $B(a, l(a))$ is used.

- *Slater*. The Slater rule produces a ranking that is inconsistent with the outcomes of as few pairwise elections as possible. That is, for a given ranking of the candidates, each pair of candidates a, b such that a is ranked higher than b, but b defeats a in their pairwise election, counts as an inconsistency, and a ranking is a Slater ranking if it minimizes the number of inconsistencies.

- *Kemeny*. This rule produces a ranking that minimizes the number of times that the ranking is inconsistent with a vote on the ranking of two candidates. That is, for a given ranking r of the candidates, each combination of a pair of candidates a, b and a vote r_a such that r ranks a higher than b, but r_i ranks b higher than a, counts as an inconsistency, and a ranking is a Kemeny ranking if it minimizes the number of inconsistencies.

We define one additional rule, the *cup* rule, which runs a single-elimination tournament to decide the winning candidate. This rule does not produce a full aggregate ranking of the candidates, and additionally requires a *schedule* for matching up the remaining candidates.

- *Cup*. This rule is defined by a balanced[1] binary tree T with one leaf per candidate, and a *schedule*, that is, an assignment of candidates to leaves (each leaf gets one candidate). Each non-leaf node is assigned the winner of the pairwise election of the node's children; the candidate assigned to the root wins. The *regular cup* rule assumes that the assignment of candidates to leaves is known by the voters before they vote. In the *randomized cup* rule, the assignment of candidates to leaves is chosen uniformly at random after the voters have voted.

Sometimes votes are *weighted*; a vote of weight K counts as K votes of weight 1. Different possible interpretations can be given to weights. They may represent the decision power of a given agent in a voting setting where not all agents are considered equal. The weight may correspond to the size of the community that the voter represents (such as the size of the state). Or, when agents vote in partisan groups (*e.g.*, in parliament), the weights may correspond to the size of the group (each group acts as one voter).

We will sometimes use the term "voting protocol" rather than "voting rule"; the meaning is roughly the same, except the word "protocol" is intended to encompass not only the mapping from rankings to outcomes (*i.e.*, aggregate rankings or winners), but also procedural aspects such as the manner in which the voters report their ranking (*e.g.*, whether all voters submit their rankings at the same time or not).

The general applicability of voting makes it an appealing approach to preference aggregation in unstructured domains. However, in more structured settings, this generality becomes a weakness, as using a voting approach does not exploit the structure of the domain. For example, many settings allow *payments* to be made by or to the agents. In principle, we can model these payments as part of the outcome, so that voter 1's vote may be something like:

[1] "Balanced" here means that the difference in depth between two leaves can be at most one.

"alternative a is chosen, voter 1 pays \$10, voter 2 pays \$5" $\succ$ "alternative b is chosen, voter 1 pays \$0, voter 2 pays \$3" $\succ$ "alternative a is chosen, voter 1 pays \$10, voter 2 pays \$6" $\succ \ldots$

Needless to say, this approach is extremely cumbersome (in principle the votes have infinite length!), and it does not exploit any of the knowledge that we have (or assumptions that we are willing to make) about how agents feel about payments. For example, we know that agents prefer smaller payments to larger ones; we may know that they do not care about other agents' payments; we may know that each dollar is as valuable as the next to an agent; *etc.*

Another drawback is that the voting approach does not allow us to make statements about how strong or weak agents' preferences over outcomes are, and hence how they feel about *distributions* over outcomes. For example, suppose an agent prefers a to b to c. Which does the agent prefer: b, or a coin flip between a and c? It is impossible to tell from the information given—we do not even know whether the agent's preference of a over b is stronger than that of b over c. Again, in principle, voters can vote over distributions over outcomes, *e.g.*:

$P(a) = .4, P(b) = .3, P(c) = .3 \succ P(a) = .5, P(b) = .2, P(c) = .3 \succ P(a) = .5, P(b) = .3, P(c) = .2 \succ \ldots$

but again this is impractical (if not impossible). Again, we can make very reasonable assumptions about agents' preferences over distributions: for example, Dutch book theorems [Mas-Colell *et al.*, 1995] suggest that agents will maximize their expected utility, because otherwise they will be susceptible to accepting a sequence of bets that is guaranteed to leave them worse off.

In the remainder of this chapter, we focus on utility-based approaches; we will return to voting (specifically, computing aggregate rankings using the Slater rule) in the next chapter, Section 3.1.

2.2 Allocation of tasks and resources

Some of the most common domains in which multiple agents' preferences must be aggregated involve the allocation of resources or tasks to the agents. I will restrict my attention to settings in which payments can be made, that is, agents can pay for resources allocated to them and tasks performed for them, or be compensated for resources they supply and tasks they perform. (Not all research on resource/task allocation makes the assumption that payments are possible: for example, Lipton *et al.* [2004] and Bouveret and Lang [2005] consider the problem of finding *envy-free* allocatons, that is, allocations under which no agent would prefer the share of another agent to its own.)

We will refer to distinct resources as *items*; the performance of a task can be thought of as an item as well, so from now on we can, without loss of generality, focus strictly on the allocation and provision of items.

Earlier, we discussed combinatorial auctions as a method for allocating a fixed set of n available items, I. Here, agent (or *bidder*) i will have a valuation function $v_i : 2^I \rightarrow \mathbb{R}$, mapping each bundle of items that could be allocated to that bidder to a real value. This is making the assumption of *no externalities*: given that a bidder does not win an item, that bidder does not care which (if any) other bidder receives the item instead. This is usually realistic, but not always: for example, a country may prefer certain other countries not to obtain certain weapons. We will discuss externalities in the

next sections. Letting $A_i \subseteq I$ denote the subset of the items that we allocate to bidder i, our goal is to find an allocation of items to the n bidders that maximizes $\sum_{i=1}^{n} v_i(A_i)$, under the constraint that for any $i \neq j$, $A_i \cap A_j = \emptyset$ (we do not award an item twice). (Note that we do not require all items to be awarded—this is known as the *free disposal assumption*.) This optimization problem is called the *winner determination problem*.

Typically, we can assume $v_i(\emptyset) = 0$, but any other restriction on the agents' valuation function will come at a loss in what the agents can express. For example, if we were to assume that $v_i(S) = \sum_{s \in S} v_i(\{s\})$, then we can no longer model complementarity (multiple items being worth more than the sum of their individual values) or substitutability (multiple items being worth less than the sum of their individual values), which is what gives combinatorial auctions their advantage over sequential or parallel auctions of individual items. On the other hand, an arbitrary valuation function requires $2^n - 1$ real values to describe, which corresponds to an infeasibly large amount of communication by a bidder if the number of items is reasonably large. This leads us to the study of *bidding languages* that the bidders can use to express their valuations. The earliest-studied language is the *OR bidding language*, in which bidders simply submit valuations for multiple bundles [Rothkopf *et al.*, 1998; DeMartini *et al.*, 1999]. An example bid is $(\{a\}, 3)$ OR $(\{b, c\}, 4)$ OR $(\{c, d\}, 2)$, which expresses that the bidder is willing to pay 3 for the bundle consisting of a alone, 4 for the bundle consisting of b and c, and 2 for the bundle consisting of c and d. In addition, *any number* of the bidder's bundles may be awarded simultaneously, at the sum of the values of the individual bundles. For example, the example bid implies that $v_i(\{a, b, c\}) = 7$. Multiple bundles cannot be awarded simultaneously if the items overlap. For example, the example bid does *not* imply that $v_i(\{b, c, d\}) = 6$. Hence, under the OR bidding language, from the perspective of winner determination, we may as well imagine that each bundle-value pair came from a separate bidder. (A bidder that is only interested in a single bundle is called a *single-minded* bidder.)

The OR bidding language is not fully expressive. For example, imagine a combinatorial auction in which two cars a and b are for sale. Now imagine a bidder that values car a at \$4,000 and car b at \$6,000. However, the bidder only requires one car, so that winning both cars would still be worth only \$6,000 to the bidder, because the bidder would simply not use car a (and we will assume that re-selling the car is impossible). This type of bidder is known as a *unit-demand* bidder, and it is impossible to capture these preferences using the OR bidding language. For example, bidding $(\{a\}, 4000)$ OR $(\{b\}, 6000)$ would imply $v_i(\{a, b\}) = 10000$. The subtitutability of the items is what causes the problem here. To address this, we can introduce *XOR-constraints* between different bundles, indicating that only one of these bundles can be accepted [Sandholm, 2002a,b]. For example, the above valuation function can be expressed by $(\{a\}, 4000)$ XOR $(\{b\}, 6000)$. Bidding with XOR constraints is fully expressive, that is, we can model any valuation function with them (even without using any ORs). To reduce the size of the bids, however, we may still wish to use ORs in addition to XORs [Nisan, 2000; Sandholm, 2002b]. The presence of XORs prevents us from pretending that each bundle-value pair was submitted by a separate bidder for the purpose of winner determination. A commonly used trick to circumvent this problem is the following: for a set of bundle-value pairs that is XORed together, create a *dummy item* that is added to all of these bundles [Fujishima *et al.*, 1999; Nisan, 2000]. Since the dummy item can only be awarded once, the dummy item effectively encodes the XOR-constraint, so that we can still imagine that each bid

comes from a separate bidder (for the purposes of winner determination).

Combinatorial auctions do not capture all possible resource/task allocation settings (even those in which payments are possible). For example, rather than having a set of items available for sale, one may instead seek to *procure* a set of items which are distributed across multiple bidders (or *suppliers*). This is especially natural in the context of task allocation, where the items to be procured correspond to tasks that must be performed. In such a setting, one can hold a *combinatorial reverse auction* [Sandholm *et al.*, 2002], in which a set of items I needs to be procured, and each bidder i has a cost function $c_i : 2^I \rightarrow \mathbb{R}$, where $c_i(S)$ indicates the cost of providing bundle S. Letting $A_i \subseteq I$ denote the subset of the items that we award to bidder i (in the sense that we require bidder i to provide them), our goal is to minimize $\sum_{i=1}^{m} c_i(A_i)$, under the constraint that $I = \bigcup_{1 \leq i \leq n} A_i$. (Again, there is a free disposal assumption here in that we allow an item to be provided by multiple bidders.)

The free disposal assumption is not always realistic: for example, one may not be able to dispose of radioactive material freely. A combinatorial forward auction *without free disposal* [Sandholm *et al.*, 2002] is exactly the same as one with free disposal, with the exception that every item must be allocated to some bidder. Here, bids with a *negative* value may be useful, as they allow us to remove some of the items—which may allow us to accept better bids for the remaining items. Similarly, a combinatorial reverse auction without free disposal is exactly the same as one with free disposal, with the exception that no additional items can be procured. Here, bids with a *negative* value may occur—the (nondisposable) item may be a liability to the bidder. In both cases, we seek to identify a subset of the bids that constitutes an exact cover of the items (no item covered more than once), and to maximize the bidders' total utility under this constraint. Therefore, the settings are technically identical, and we can without loss of generality restrict our attention to forward auctions without free disposal.

In general, it is not the case that either all agents are seeking only to procure items, or all agents are seeking only to provide items. Rather, some agents may be seeking to procure items; others, to provide items; and yet others, to do both simultaneously. This leads to the model of a *combinatorial exchange* [Sandholm *et al.*, 2002], in which there is a set of m items I for sale, and bidder i has a valuation functions $v_i : \mathbb{Z}^m \rightarrow \mathbb{R}$. Here, $v_i(\lambda_1, \ldots, \lambda_m)$ is bidder i's value for receiving λ_j units of item j. (If λ_j is negative, that means that the bidder is providing units of that item.)

It should be noted that the notion of bidding for multiple units of the same item can be applied to combinatorial auctions and reverse auctions as well. However, there it is not strictly necessary, in the sense that we can re-model a combinatorial (reverse) auction with multiple units as one with single units, by describing each individual unit of an item that is for sale or to be procured as a separate item. The "sameness" of some of these new items will then be implied by the fact that the bidders' valuation or cost functions treat these items symmetrically. (This may, however, be grossly inefficient from a representational and computational standpoint.) In an exchange, however, the number of units of each item that is for sale/to be procured is not known *ex ante*, which prevents this trick.

We will return to combinatorial auctions (specifically, to the complexity of the winner determination problem) in the next chapter, Section 3.2. In the remainder of this chapter, we will focus on settings where an agent's utility depends on more than his own items, tasks, and payments—that is,

we will drop the *no externalities* assumption.

2.3 Donations to (charitable) causes

When money is donated to a charitable (or other) cause (hereafter simply referred to as a *charity*), often the donating party gives *unconditionally*: a fixed amount is transferred from the donator to the charity, and none of this transfer is contingent on other events—in particular, it is not contingent on the amount given by other parties. Indeed, this is currently often the only way to make a donation, especially for small donating parties such as private individuals. However, when multiple parties support the same charity, each of them would prefer to see the others give more rather than less to this charity. In such scenarios, it is sensible for a party to use its contemplated donation to induce the others to give more. This is done by making the donation conditional on the others' donations. The following example will illustrate this, and show that the donating parties as well as the charitable cause may simultaneously benefit from the potential for such negotiation.

Suppose we have two parties, 1 and 2, who are both supporters of charity A. To either of them, it would be worth \$0.75 if A received \$1. It follows neither of them will be willing to give unconditionally, because \$0.75 $<$ \$1. However, if the two parties draw up a contract that says that they will each give \$0.5, both the parties have an incentive to accept this contract (rather than have no contract at all): with the contract, the charity will receive \$1 (rather than \$0 without a contract), which is worth \$0.75 to each party, which is greater than the \$0.5 that that party will have to give. Effectively, each party has made its donation conditional on the other party's donation, leading to larger donations and greater happiness to all parties involved.[2]

One method that is often used to effect this is to make a *matching offer*. Examples of matching offers are: "I will give x dollars for every dollar donated.", or "I will give x dollars if the total collected from other parties exceeds y." In our example above, one of the parties can make the offer "I will donate \$0.5 if the other party also donates at least that much", and the other party will have an incentive to indeed donate \$0.5, so that the total amount given to the charity increases by \$1. Thus this matching offer implements the contract suggested above. As a real-world example, the United States government has authorized a donation of up to \$1 billion to the Global Fund to fight AIDS, TB and Malaria, under the condition that the American contribution does not exceed one third of the total—to encourage other countries to give more [Tagliabue, 2003].

However, there are several severe limitations to the simple approach of matching offers as just described.

1. It is not clear how two parties can make matching offers where each party's offer is stated in terms of the amount that the other pays. (For example, it is not clear what the outcome should be when both parties offer to match the other's donation.) Thus, matching offers can

[2]The preferences given in this example do not consider the possibility that an agent's utility depends not only on how much the charity receives but also on the extent to which that agent feels responsible for it. For example, an agent may feel better about the scenario in which the agent gives \$1 to a charity than about the scenario in which the agent loses \$1 gambling and another agent gives \$1 to the charity. However, we will not consider such preferences and assume that an agent only cares about the final amount received by each charity (as well as about the agent's own final budget).

only be based on payments made by parties that are giving unconditionally (not in terms of a matching offer)—or at least there can be no circular dependencies.[3]

2. Given the current infrastructure for making matching offers, it is impractical to make a matching offer depend on the amounts given to *multiple* charities. For instance, a party may wish to specify that it will pay \$100 given that charity A receives a total of \$1000, but that it will also count donations made to charity B, at half the rate. (Thus, a total payment of \$500 to charity A combined with a total payment of \$1000 to charity B would be just enough for the party's offer to take effect.)

In contrast, in this section we propose a new approach where each party can express its relative preferences for different charities, and make its offer conditional on its own appreciation for the vector of donations made to the different charities. Moreover, the amount the party offers to donate at different levels of appreciation is allowed to vary arbitrarily (it does need to be a dollar-for-dollar (or n-dollar-for-dollar) matching arrangement, or an arrangement where the party offers a fixed amount provided a given (strike) total has been exceeded). Finally, there is a clear interpretation of what it means when multiple parties are making conditional offers that are stated in terms of each other. Given each combination of (conditional) offers, there is a (usually) unique solution which determines how much each party pays, and how much each charity is paid.

In short, expressive preference aggregation for donations to charities is a new way in which electronic commerce can help the world. A web-based implementation of the ideas described in this section can facilitate voluntary reallocation of wealth on a global scale.

2.3.1 Definitions

Throughout, we will refer to the offers that the donating parties make as *bids*, and to the donating parties as *bidders*. In our bidding framework, a bid will specify, for each vector of total payments made to the charities, how much that bidder is willing to contribute. (The contribution of this bidder is also counted in the vector of payments—so, the vector of total payments to the charities represents the amount given by *all* donating parties, not just the ones other than this bidder.) The bidding language is expressive enough that no bidder should have to make more than one bid. The following definition makes the general form of a bid in our framework precise.

Definition 1 *In a setting with m charities $c_1, c_2, \ldots, c_m$, a* bid *by bidder b_j is a function $v_j : \mathbb{R}^m \rightarrow \mathbb{R}$. The interpretation is that if charity c_i receives a total amount of π_{c_i}, then bidder j is willing to donate (up to) $v_j(\pi_{c_1}, \pi_{c_2}, \ldots, \pi_{c_m})$.*

We now define possible outcomes in our model, and which outcomes are valid given the bids that were made.

Definition 2 *An* outcome *is a vector of payments made by the bidders $(\pi_{b_1}, \pi_{b_2}, \ldots, \pi_{b_n})$, and a vector of payments received by the charities $(\pi_{c_1}, \pi_{c_2}, \ldots, \pi_{c_m})$. A* valid *outcome is an outcome where*

[3]Typically, larger organizations match offers of private individuals. For example, the American Red Cross Liberty Disaster Fund maintains a list of businesses that match their customers' donations [Goldburg and McElligott, 2001].

1. $\sum_{j=1}^{n} \pi_{b_j} \geq \sum_{i=1}^{m} \pi_{c_i}$ *(at least as much money is collected as is given away);*

2. *For all* $1 \leq j \leq n$, $\pi_{b_j} \leq v_j(\pi_{c_1}, \pi_{c_2}, \ldots, \pi_{c_m})$ *(no bidder gives more than she is willing to).*

Of course, in the end, only one of the valid outcomes can be chosen. We choose the valid outcome that maximizes the *objective* that we have for the donation process.

Definition 3 *An* objective *is a function from the set of all outcomes to* $\mathbb{R}$.[4] *After all bids have been collected, a valid outcome will be chosen that maximizes this objective.*

One example of an objective is *surplus*, given by $\sum_{j=1}^{n} \pi_{b_j} - \sum_{i=1}^{m} \pi_{c_i}$. The surplus could be the profits of a company managing the expressive donation marketplace; but, alternatively, the surplus could be returned to the bidders, or given to the charities. Another objective is *total amount donated*, given by $\sum_{i=1}^{m} \pi_{c_i}$. (Here, different weights could also be placed on the different charities.)

2.3.2 A simplified bidding language

Specifying a general bid in our framework (as defined above) requires being able to specify an arbitrary real-valued function over $\mathbb{R}^m$. Even if we restricted the possible total payment made to each charity to the set $\{0, 1, 2, \ldots, s\}$, this would still require a bidder to specify $(s+1)^m$ values. Thus, we need a bidding language that will allow the bidders to at least specify *some* bids more concisely. We will specify a bidding language that only represents a subset of all possible bids, which can be described concisely.[5]

To introduce our bidding language, we will first describe the bidding function as a composition of two functions; then we will outline our assumptions on each of these functions. First, there is a *utility function* $u_j : \mathbb{R}^m \to \mathbb{R}$, specifying how much bidder j appreciates a given vector of total donations to the charities. (Note that the way we define a bidder's utility function, it does not take the payments the bidder makes into account.) Then, there is a *donation willingness* function $w_j : \mathbb{R} \to \mathbb{R}$, which specifies how much bidder j is willing to pay given her utility for the vector of donations to the charities. We emphasize that this function does *not* need to be linear, so that utilities should not be thought of as expressible in dollar amounts. (Indeed, when an individual is donating to a large charity, the reason that the individual donates only a bounded amount is typically not decreasing marginal value of the money given to the charity, but rather that the marginal value of a dollar to the bidder herself becomes larger as her budget becomes smaller.) So, we have $w_j(u_j(\pi_{c_1}, \pi_{c_2}, \ldots, \pi_{c_m})) = v_j(\pi_{c_1}, \pi_{c_2}, \ldots, \pi_{c_m})$, and we let the bidder describe her functions u_j and w_j separately. (She will submit these functions as her bid.)

[4] In general, the objective function may also depend on the bids, but the objective functions that we consider do not depend on the bids. The techniques presented in this dissertation will typically generalize to objectives that take the bids into account directly.

[5] Of course, our bidding language can be trivially extended to allow for fully expressive bids, by also allowing bids from a fully expressive bidding language, in addition to the bids in our bidding language.

Our first restriction is that the utility that a bidder derives from money donated to one charity is *independent* of the amount donated to another charity. Thus, $u_j(\pi_{c_1}, \pi_{c_2}, \ldots, \pi_{c_m}) = \sum_{i=1}^{m} u_j^i(\pi_{c_i})$. (We observe that this does *not* imply that the bid function v_j decomposes similarly, because of the nonlinearity of w_j.) Furthermore, each u_j^i must be piecewise linear. An interesting special case which we will study is when each u_j^i is a line: $u_j^i(\pi_{c_i}) = a_j^i \pi_{c_i}$. This special case is justified in settings where the scale of the donations by the bidders is small relative to the amounts the charities receive from other sources, so that the marginal use of a dollar to the charity is not affected by the amount given by the bidders.

The only restriction that we place on the payment willingness functions w_j is that they are piecewise linear. One interesting special case is a *threshold bid*, where w_j is a step function: the bidder will provide t dollars if her utility exceeds s, and otherwise 0. Another interesting case is when such a bid is *partially acceptable*: the bidder will provide t dollars if her utility exceeds s; but if her utility is $u < s$, she is still willing to provide $\frac{ut}{s}$ dollars.

One might wonder why, if we are given the bidders' utility functions, we do not simply maximize the sum of the utilities rather than surplus or total donated. There are several reasons. First, because affine transformations do not affect utility functions in a fundamental way, it would be possible for a bidder to inflate her utility by changing its units, thereby making her bid more important for utility maximization purposes. Second, a bidder could simply give a payment willingness function that is 0 everywhere, and have her utility be taken into account in deciding on the outcome, in spite of her not contributing anything.

2.3.3 Avoiding indirect payments

In an initial implementation, the approach of having donations made out to a center, and having a center forward these payments to charities, may not be desirable. Rather, it may be preferable to have a *partially decentralized* solution, where the donating parties write out checks to the charities directly according to a solution prescribed by the center. In this scenario, the center merely has to verify that parties are giving the prescribed amounts. Advantages of this include that the center can keep its legal status minimal, as well as that we do not require the donating parties to trust the center to transfer their donations to the charities (or require some complicated verification protocol). It is also a step towards a fully decentralized solution, if this is desirable.

To bring this about, we can still use the approach described earlier. After we clear the market in the manner described before, we know the amount that each donator is supposed to give, and the amount that each charity is supposed to receive. Then, it is straightforward to give some specification of who should give how much to which charity, that is consistent with that clearing.

Nevertheless, with this approach, a bidder may have to write out a check to a charity that she does not care for at all. (For example, an environmental activist who was using the system to increase donations to a wildlife preservation fund may be required to write a check to a group supporting a right-wing political party.) This is likely to lead to complaints and noncompliance with the clearing. We can address this issue by letting each bidder specify explicitly (before the clearing) which charities she would be willing to make a check out to. These additional constraints, of course, may change the optimal solution.

The setting of expressive preference aggregation for donations to charities is a special case of a more general phenomenon, namely that agents' actions may indirectly affect other agents' utilities. We study this more general setting in the next section. We will return to the more specific setting of expressive preference aggregation for donations to charities (specifically, to the complexity of computing optimal outcomes in this setting) in the next chapter, Section 3.3.

2.4 Public goods and externalities

A pervasive assumption in the research on combinatorial auctions and exchanges has been that there are *no allocative externalities*: no agent cares what happens to an item unless that agent itself receives the item. This is insufficient to model situations where there are certain items (such as nuclear weapons) that are such that bidders who do not win the item still care which other bidder wins it [Jehiel and Moldovanu, 1996]. More generally, there are many important preference aggregation settings where decisions taken by a few agents may affect many other agents. For example, many agents may benefit from one agent taking on a task such as building a bridge (and the extent of their benefit may depend on how the bridge is built, for example, on how heavy a load it can support). Alternatively, if a company reduces its pollution level, many individuals may benefit, even if they have nothing to do with the goods that the company produces. A decision's effect on an otherwise uninvolved agent is commonly known as an *externality* [Mas-Colell *et al.*, 1995]. In designing a good preference aggregation protocol, externalities must be taken into account, so that (potentially complex) arrangements can be made that are truly to every agent's benefit.

In this section, we define a representation for combinatorial preference aggregation settings with externalities. We will mostly focus on restricted settings that cannot model *e.g.* fully general combinatorial auctions and exchanges, so that we do not inherit all of the complexities from those settings.

We formalize the problem setting as follows.

Definition 4 *In a* setting with externalities*, there are n agents $1, 2, \ldots, n$; each agent i controls m_i variables $x_i^1, x_i^2, \ldots, x_i^{m_i} \in \mathbb{R}^{\geq 0}$; and each agent i has a utility function $u_i : \mathbb{R}^M \rightarrow \mathbb{R}$ (where $M = \sum_{j=1}^{n} m_j$). (Here, $u_i(x_1^1, \ldots, x_1^{m_1} \ldots, x_n^1, \ldots, x_n^{m_n})$ represents agent i's utility for any given setting of the variables.)*

In general, one can also impose constraints on which values for $(x_i^1, \ldots, x_i^{m_i})$ agent i can choose, but we will refrain from doing so in this section. (We can effectively exclude certain settings by making the utilities for them very negative.) We say that the *default outcome* is the one where all the x_i^j are set to 0,[6] and we require without loss of generality that all agents' utilities are 0 at the default outcome. Thus, the participation constraint states that every agent's utility should be nonnegative.

[6]This is without loss of generality because the variables x_i^j can be used to represent the *changes* in the real-world variables relative to the default outcome. If these changes can be both positive and negative for some real-world variable, we can model this with two variables $x_i^{j_1}, x_i^{j_2}$, the difference between which represents the change in the real-world variable.

Without any restrictions placed on it, this definition is very general. For instance, we can model a (multi-item, multi-unit) combinatorial exchange with it. Recall that in a combinatorial exchange, each agent has an initial endowment of a number of units of each item, as well as preferences over endowments (possibly including items not currently in the agent's possession). The goal is to find some reallocation of the items (possibly together with a specification of payments to be made and received) so that no agent is left worse off, and some objective is maximized under this constraint. We can model this in our framework as follows: for each agent, for each item in that agent's possession, for each other agent, let there be a variable representing how many units of that item the former agent transfers to the latter agent. (If payments are allowed, then we additionally need variables representing the payment from each agent to each other agent.) We note that this framework allows for *allocative externalities*, that is, for the expression of preferences over which of the other agents receives a particular item.

Of course, if the agents can have nonlinear preferences over bundles of items (there are complementarities or substitutabilities among the items), then (barring some special concise representation) specifying the utility functions requires an exponential number of values.[7] We need to make some assumption about the structure of the utility functions if we do not want to specify an exponential number of values. For the most part, we make the following assumption, which states that the effect of one variable on an agent's utility is independent of the effect of another variable on that agent's utility. We note that this assumption disallows the model of a combinatorial exchange that we just gave, unless there are no complementarities or substitutabilities among the items. This is not a problem insofar as our primary interest here is not so much in combinatorial exchanges as it is in more natural, simpler externality problems such as aggregating preferences over pollution levels. We note that this restriction makes the hardness results on outcome optimization that we present in the next chapter (Section 3.4) much more interesting (without the restriction, the results would have been unsurprising given known hardness results in combinatorial exchanges). However, for some of our positive results we will actually not need the assumption, for example for convergence results for an algorithm that we will present.

Definition 5 u_i decomposes (across variables) *if* $u_i(x_1^1, \ldots, x_1^{m_1}, \ldots, x_n^1, \ldots, x_n^{m_n}) = \sum_{k=1}^{n} \sum_{j=1}^{m_k} u_i^{k,j}(x_k^j)$.

When utility functions decompose, we will sometimes be interested in the special cases where the $u_i^{k,j}$ are step functions (denoted $\delta_{x \geq a}$, which evaluates to 0 if $x < a$ and to 1 otherwise), or piecewise constant functions (linear combinations of step functions).[8]

In addition, we will focus strictly on settings where the higher an agent sets its variables, the worse it is for itself. We will call such settings *concessions settings*. So, if there is no preference aggregation, each agent will selfishly set all its variables to 0 (the *default* outcome).

[7] Thus, the fact that finding a feasible solution for a combinatorial exchange is NP-complete [Sandholm *et al.*, 2002] does not imply that finding a feasible solution in our framework is NP-complete, because there is an exponential blowup in representation.

[8] For these special cases, it may be conceptually desirable to make the domains of the variables x_i^j discrete, but we will refrain from doing so in this dissertation for the sake of consistency.

Definition 6 *A* concessions setting *is a setting with externalities for which for any* $(x_1^1, \ldots, x_1^{m_1}, \ldots, x_n^1, \ldots, x_n^{m_n}) \in \mathbb{R}^M$*, for any* $i, 1 \leq j \leq m_i$*, and for any* $\hat{x}_i^j > x_i^j$*, we have* $u_i(x_1^1, \ldots, x_1^{m_1}, \ldots, \hat{x}_i^j, \ldots, x_n^1, \ldots, x_n^{m_n}) \leq u_i(x_1^1, \ldots, x_1^{m_1}, \ldots, x_i^j, \ldots, x_n^1, \ldots, x_n^{m_n})$.

In parts of this dissertation, we will be interested in the following additional assumption, which states that the higher an agent sets its variables, the better it is for the others. (For instance, the more a company reduces its pollution, the better it is for all others involved.)

Definition 7 *A concessions setting has* only negative externalities *if for any* $(x_1^1, \ldots, x_1^{m_1}, \ldots, x_n^1, \ldots, x_n^{m_n}) \in \mathbb{R}^M$*, for any* $i, 1 \leq j \leq m_i$*, for any* $\hat{x}_i^j > x_i^j$*, and for any* $k \neq i$*,* $u_k(x_1^1, \ldots, x_1^{m_1}, \ldots, \hat{x}_i^j, \ldots, x_n^1, \ldots, x_n^{m_n}) \geq u_k(x_1^1, \ldots, x_1^{m_1}, \ldots, x_i^j, \ldots, x_n^1, \ldots, x_n^{m_n})$.

We define *trivial* settings of variables as settings that are indistinguishable from setting them to 0.

Definition 8 *The value* r *is* trivial for *variable* x_i^j *if it does not matter to anyone's utility function whether* x_i^j *is set to* r *or to* 0*. (That is, for any* $x_1^1, \ldots, x_1^{m_1}, \ldots, x_i^{j-1}, x_i^{j+1}, \ldots, x_n^1, \ldots, x_n^{m_n}$*, and for any* k*, we have* $u_k(x_1^1, \ldots, x_1^{m_1}, \ldots, x_i^{j-1}, r, x_i^{j+1}, \ldots, x_n^1, \ldots, x_n^{m_n}) = u_k(x_1^1, \ldots, x_1^{m_1}, \ldots, x_i^{j-1}, 0, x_i^{j+1}, \ldots, x_n^1, \ldots, x_n^{m_n})$*. A setting of all the variables is* trivial *if each variable is set to a trivial value.*

Say that an outcome is *feasible* if no agent prefers the default outcome to it (and would therefore try to block the preference aggregation process). Our goal is to find a good feasible outcome. As in the setting of expressive preference aggregation for donations to charities, there can be multiple objectives. Interesting objectives include social welfare and total concessions (the sum of the variables). A more modest, but nevertheless interesting goal is to simply discover a nontrivial feasible solution. We will study how hard these problems are computationally in the next chapter, Section 3.4.

2.5 Summary

In this chapter, we reviewed four different settings for preference aggregation. We also reviewed corresponding languages in which agents can express their preferences, and criteria according to which the outcome can be selected.

In Section 2.1, we reviewed *voting settings*, where agents (or voters) can rank the alternatives (or candidates) in any order, and the winner or aggregate ranking of the alternatives is chosen based on the rankings (or votes) submitted by the voters. We reviewed some basic concepts from social choice theory (Condorcet cycles, Arrow's impossibility result, and single-peaked preferences), as well as a number of specific rules for choosing the outcome based on the submitted votes.

In Section 2.2, we reviewed allocation of tasks and resources, using combinatorial auctions, reverse auctions, and exchanges. We defined the winner determination problem and reviewed basic bidding languages, including OR- and XOR-based languages.

In Section 2.3, we introduced a new application: expressive preference aggregation for donations to charities. The idea here is that when donating money to a charity, it is possible to use

the contemplated donation in negotiation to induce other parties interested in the charity to donate more. We introduced a bidding language for expressing very general types of matching offers over multiple charities, and formulated the corresponding clearing problem (deciding how much each bidder pays, and how much each charity receives).

Finally, in Section 2.4, we introduced a framework for negotiating in general settings with externalities. Again, we introduced a language in which agents can express their preferences, and criteria according to which to select the final outcome.

So far, we have not yet assessed how difficult it is computationally to solve the outcome optimization problem, *i.e.* to select the optimal outcome according to the given criteria based on the preferences submitted by the agents, in any of these settings. This is the topic of the next chapter.