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