Refinements and randomised versions of some tournament solutions

Justin Kruger and Stéphane Airiau

Abstract

We consider voting rules that are based on the majority graph. Such rules typically output large sets of winners. Our goal is to investigate a general method which leads to refinements of such rules. In particular, we use the idea of parallel universes, where each universe is connected with a permutation over alternatives. The permutation allows us to construct resolute voting rules (i.e. rules that always choose unique winners). Such resolute rules can be constructed in a variety of ways: we consider using binary voting trees to select a single alternative. In turn this permits the construction of neutral rules that output the set the possible winners of every parallel universe. The question of which rules can be constructed in this way has already been partially studied under the heading of agenda implementability. We further propose a randomised version in which the probability of being the winner is the ratio of universes in which the alternative wins. We also investigate (typically novel) rules that elect the alternatives that have maximal winning probability. These rules typically output small sets of winners, thus provide refinements of known tournament solutions.

keywords:
tournament probabilistic rules refinements Condorcet consistency

1 Introduction

In general, social choice theory studies the problem of making group decisions: the problem of selecting a single alternative from a set of alternatives, given that different members of the group have different opinions and preferences. Stated as such this is a rather vague problem. One attempt to make it more tractable is to restrict attention to two alternatives at a time. Such a focus leads to structures called tournaments, which have been objects of purely mathematical studies for a long time. For social choice theoretic purposes a tournament can be defined in the following manner: if a majority of people in the society prefer a to b, then there is a directed edge from a to b. The hope is that using a tournament to determine the selected alternative will allow us to fairly select the best alternative in a consistent and transparent manner.

There are many methods (which we will call rules) designed to select alternatives from a tournament. These include, but are not restricted to: Copeland, the Top Cycle, Banks, Slater and the Markov solution concept. All of these satisfy what is known as the Condorcet Criterion, which, roughly speaking, requires that an obviously best alternative is selected. More precisely: if a single alternative wins every pairwise competition between itself and any other candidate, then any reasonable rule will select this alternative.

What about when a tournament has no obvious winner? Another, less fortunate, trait often exhibited by rules based on tournaments is irresoluteness. Such rules often cannot decide between alternatives; instead of selecting a unique alternative they output a set of multiple alternatives. Resoluteness (always selecting a single alternative) is often required, either for actual implementations or to facilitate particular analyses of social choice rules. Thus tournament rules are typically equipped with an exogeneous tie-breaking method that is applied after the rule. In terms of fairness using such a tie-breaker violates the highly desirable property of neutrality; it is no longer the case that all the alternatives are considered equal.

So far we have only discussed social choice theoretic issues concerning methods for selecting from tournaments. However such rules are also interesting from an algorithmic standpoint. One interesting result is that it can be hard to compute the full set of selected alternatives while at the same time easy to find some winner. Rules based on tournaments are not alone in having this gap; social choice functions based on some sort of parallel computation often seem to have this characteristic. Typically, for a parallel computation rule, a trade-off must be made between neutrality on the one hand, and resoluteness and tractability on the other.

In this paper we ignore the question of tractability and focus on the question of resoluteness. We know that full resoluteness is not achievable at the same time as neutrality. Instead we ask whether it can be possible that the rule can be *more* resolute while remaining neutral. We can pose this question in two ways: (i) are there more cases where a single winner is selected? and (ii) in those cases where multiple winners are still selected, is the set smaller than before? The methods used to generate rules of this type also lead to a natural definition of randomised rules. We will also define and analyse these rules.

1.1 Previous work

There is a large literature on tournaments ranging from purely mathematical [15] to explicitly social choice theoretic. Definitions of the rules described in the introduction can be found [13] and [4], either of which provide extensive general treatments of tournaments from the social choice theoretic side.

The particular tournament rules that we are interested in concern tree structures, of which one strand of study originates from questions concerning voting by agenda. Perhaps the first significant result is the implementation of what became known as the Banks solution concept [14, 2]. This in turn lead to more general questions of implementability: attempts to find the rules implementable by individual resolute trees were given by Srivastava and Trick [17] and continued by Trick [19]. To some extent this issue is settled by Horan [9], who went on to give full sufficient and necessary conditions for irresolute tournament rules [10].

The Banks solution concept provides a good example of the gap in difficulty between calculating all and calculating some winners: Woeginger [20] showed that the full set is NP-hard, to which Hudry [11] replied noting there is a greedy algorithm for calculating some Banks winner. These results are implicitly based upon the "parallel" nature of the Banks solution concept. Such implicit treatment of the idea of parallel computations can be traced back to Tideman's Ranked Pair rule [18], but seems to have been first explicitly called such by Conitzer et al. [6] who investigate the single transferrable vote rule. Freeman et al. [8] continued this study in a similar direction, while Brill and Fischer [5] applied a similar explicit treatment of parallel universes to Ranked Pairs.

1.2 Outline

In the following section we collect and unify those definitions from the literature that we will require, in particular definitions concerning resolute rules determined by trees. In the section after this we apply parallel universes terminology to these voting trees, thereby defining two deterministic rules and one probabilistic rule. In the penultimate section we investigate which different properties apply to the different versions of these rules and ask to what extent we have defined refinements of known rules. The final selection summarises our results and indicates future directions.

2 Definitions

In this section we give preliminary definitions. All of these are given with reference to a set X of m alternatives. We will label specific alternatives of X as $1, 2, 3, \dots$ and refer to arbitrary elements as $a, b, c, \dots$

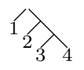
The typical comparison method between two alternatives is majority voting: alternative a defeats alternative b if a wins in a pairwise majority election. We will refer to this as the domination relation, i.e. "a dominates b" expresses the relation aTb. If we assume that such a relation holds between every pair of alternatives, we end up with a tournament. Formally, a *tournament* is a trichotomous[1] binary relation T over some set X. Perhaps the smallest interesting example involves four alternatives.

Example 1. *Let T and T' be tournaments with aTb or $aT'b$ if there is an arrow from a to b in the following respective graphs.*

$$T = \begin{tikzcd} 4 \arrow[d] \arrow[dr] & 3 \arrow[l] \arrow[dl, thick] \\ 1 \arrow[r] & 2 \arrow[u] \end{tikzcd} \qquad\qquad T' = \begin{tikzcd} 4 \arrow[d] \arrow[dr] & 3 \arrow[l] \\ 1 \arrow[r] \arrow[ur, thick] & 2 \arrow[u] \end{tikzcd}$$

Note that in the above example we can obtain T' from T by switching the arrow between 1 and 3. In general we will write $T_{\langle a,b\rangle}$ for the tournament obtained by reversing the relation between a and b in T.

A *tournament function* is a function from the set of all tournaments over some set X to subsets of X. Thus a tournament function selects a set of winners from a tournament. One common way to define tournament functions is using binary trees. Indeed, for the rest of the paper we will focus on rules defined in terms of such trees. We will use two kinds of description for trees: either a graphical or a compact representation, for example:



1(2(34))

The compact representation of the tree is a left-associative word, where parantheses (i) are assumed to exist between the two leftmost elements in a triple and (ii) indicate that two nodes are siblings. This particular tree structure has gone under multiple names: it is equivalent to the sincere simple agenda [13], but has also been called the voting caterpillar [7]. We will call it the simple tree, and give the following recursive definition (applicable to any number of alternatives).

Simple tree $\mathsf{st}(1, 2, \ldots, n) = 1\ (\mathsf{st}(2, \ldots, n))$

We now define a tournament function based on a tree ω over the set of candidate X, which we note $[\![\omega]\!]$. The tournament T provides the dominance relation over the set of candidates X and the tree describes an order of the pairwise competitions between the candidates, so $[\![\omega]\!](T)$ is the winner of the tournament T. We use the natural way to determine the winner: we recursively determine the label of a parent as whichever child dominates the other child with reference to the given tournament (unless both children are the same in which case this alternative is also set as the parent). The selected winner is then the alternative at the root node. Formally, for a tree ω, the tournament function $[\![\omega]\!]$ has the following recursive definition:

$$[\![xy]\!](T) = \begin{cases} x & \text{if } xTy \\ y & \text{otherwise} \end{cases}$$

$$[\![\omega(\omega')]\!](T) = [\![\ [\![\omega]\!](T)\ [\![\omega']\!](T)\ ]\!]\,(T)$$

It can be verified that for the tournaments in Example 1, $[\![1(2(34))]\!](T) = [\![1(2(34))]\!](T') = 1$.

Clearly, any tournament function defined by a binary tree will always select a single alternative: we say such a tournament function is *resolute*. Resoluteness is incompatible with *neutrality*, which requires that permuting the alternatives in the tournament results in an identical permutation of the alternatives in the output set. Neutrality is one of the key properties that we desire of our rules.

[1]Recall a binary relation $R \subseteq X \times X$ *trichotomous* if for all $a, b \in X$, either aTb, bTa or $a = b$.

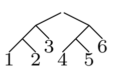
Another key property is the *Condorcet Criterion*, which requires that if there is a alternative that pairwise defeats all other alternatives, this alternative is selected. It is easy to see that only *complete* binary trees, those that have every alternative appearing as *some* leaf, satisfy the Condorcet Criterion (proofs are given in [19, 9]). Thus the *two-leaf tree* violates this when $m > 2$.

Two-leaf tree $\mathsf{tt}(1, 2, \ldots) = 12$.

Also desirable is *monotonicity*, which requires that whenever a winner is reinforced it does not become a loser. Reinforcing a winner here means making it dominate more alternatives than before, while not changing the domination relations between any other alternatives. Formally, if T is a tournament such that bTa, and a is selected by the tournament function, then a should also be selected by $T_{\langle a,b\rangle}$. Notice that alternative 1 has been reinforced between T and T' in Example 1, and that the simple tree outputs 1 for both of these tournaments; in fact, the simple tree is monotonic. Further, all *non-repetitive* binary trees with no repeated alternatives in their leaves are monotonic (a proof is given, for example, in [13]). Another example of such a non-repetitive tree is the "fair" or "balanced" voting tree [12] for which the height is minimised.

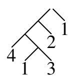
Balanced tree $\mathsf{ft}(1, \ldots, \lceil \frac{n}{2} \rceil, \lceil \frac{n}{2} \rceil + 1, \ldots, n) = \mathsf{ft}(1, \ldots, \lceil \frac{n}{2} \rceil) \; \big(\mathsf{ft}(\lceil \frac{n}{2} \rceil + 1, \ldots, n) \big)$

We note that if $\log_2 |X|$ is not an integer there are multiple non-repetitive tree structures that have minimum height. However, our particular implementation is the perhaps the most natural, as it also minimises the difference in the amount of nodes in the left and right subtrees of any particular node. For example[2]:

$\mathsf{ft}(1, 2, 3, 4, 5, 6) =$ 

123(456)

Other trees may not be monotonic. We give a minimal counter example: let ω be the following binary tree.

$\omega =$ 

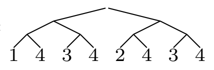
4(13)21

It can be verified that for T and T' of Example 1, we have $[\![\omega]\!](T) = 1$, but $[\![\omega]\!](T') = 3$.

However, some repetitive trees are monotonic. Sophisticated voting on the simple agenda results in an outcome produced by the following so-called Banks tree, named for one of its early investigators [2].

Banks tree $\mathsf{bt}(1, 2, \ldots, n) = \mathsf{bt}(1, 3, \ldots, n) \; (\mathsf{bt}(2, 3, \ldots, n))$

For four alternatives this produces the following tree.

$\mathsf{bt}(1, 2, 3, 4) =$ 

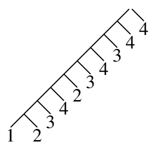
14(34)(24(34))

The fact that this is monotonic is well known; we give an example proof later (Proposition 4.17). Another monotonic (by Proposition 4.18 to come) repetitive tree is the following, which adapts the iterative Condorcet rule described by Altman and Klienberg [1] into a binary tree rule.

Iterative Condorcet tree $\mathsf{it}(1, 2, \ldots, n) = 12 \ldots n \; \mathsf{it}(2, \ldots n)$

Intuitively, given a fixed ordered over the candidates, this rule successively removes alternatives from the tournament until the smaller tournament has a Condorcet winner. This Condorcet winner is then selected. To our knowledge this has algorithm has not been implemented as a tree before.

[2]Note that by left associativity many of the brackets in the compact representation are "left out".

it$(1,2,3,4) =$ 

1234234344

Neutrality allows for a direct application of a tournament function on one set of alternatives to a tournament function on another set of alternatives of the same size, with the aid of any bijection between the two sets of alternatives. This provides the first step towards extending tournament functions to apply to any set of alternatives. A *tournament solution* is a family of neutral tournament functions that can be applied to tournaments over any set. Thus a tournament function is restricted to tournaments over some particular set (and can be non-neutral), whereas tournament solutions are applicable to tournaments over any set (and are necessarily neutral). Properties that are applicable to tournament functions can be said to hold for tournament solutions if they hold for every tournament function in the family.

For a tournament T over X, its *restriction* to $Y \subseteq X$ is the tournament $T_Y = \{(x,y) \in T \mid x,y \in Y\}$. This subtournament T_Y is further a *component* if all its alternatives have the same relation to any element outside the component. Formally we require that for all $a,b \in Y$ and $c \in X \backslash Y$, aTc iff bTc. If a tournament has components, if can be sensibly split into smaller parts. Thus a *decomposition* of a tournament is a division of the tournament into components T_{X_i}, fully written as $(T^*, T_{X_1}, \ldots, T_{X_k})$ such that the X_is are (i) pairwise disjoint, (ii) cover the set X, and (iii) form components when the tournament T is restricted to them. The tournament T^* is called the *summary* of the decomposition: a tournament over $\{1,\ldots,k\}$ such that iT^*j iff xTy for all $x \in X_i$ and $y \in X_j$.

Example 2. *Consider the tournament T of Example 1. This can be decomposed into $(T^*, T_{\{1,4\}}, T_{\{3\}}, T_{\{2\}})$ such that*

$$T^* = \begin{tikzcd} & \{3\} \arrow[dl] \\ \{1,4\} \arrow[dr] & \\ & \{2\} \arrow[uu] \end{tikzcd} \qquad\qquad T_{\{1,4\}} = \begin{tikzcd} 4 \arrow[d] \\ 1 \end{tikzcd}$$

Weak composition consistency requires that local changes to a component (i) do not change the winners on other components and (ii) do not change the fact that either no element or some element wins on the component itself. Formally, given a component Y and $x,y \in Y$, (i) $F(T) \cap (X \backslash Y) = F(T_{\langle x,y \rangle}) \cap (X \backslash Y)$ and (ii) $F(T) \cap Y \neq \emptyset \leftrightarrow F(T_{\langle x,y \rangle}) \cap Y \neq \emptyset$.

Weak composition consistency is a weakening of *composition consistency*, which states that the winners of the overall tournament should be winners of the winners in a decomposition: one can first determine the winners of the summary and then determine the winners of these winning tournaments. Unlike weak composition consistency this can only apply to tournament solutions; formally for any decomposition $T = (T^*, T_{X_1}, \ldots, T_{X_k})$, $a \in S(T)$ iff $S(T^*) = i$ where $a \in X_i$ and $a \in S(T_{X_i})$. For any tournament solution to which composition consistency applies, the consituent tournament functions will satisfy weak composition consistency.

The final property of this section is *stability*. According to [4], stability requires that a set is chosen from two different sets of alternatives if and only if it is chosen from the union of these sets. A tournament solution S is *stable* if for all tournaments function and for all nonempty subsets Y, Z of X and $W \subseteq Y \cap Z$, $W = S(Y) = S(Z)$ if and only if $W = S(Y \cap Z)$.

3 From parallel universe to randomised rules and refinements

In this section we move from resolute rules defined on trees to neutral rules. To do this we consider permutations over the set of alternatives. Such a permutation can be applied to the leaves of a voting tree. The winner of this permuted tree is then called a co-winner. It is well known that union of the outcomes over all permutations of the simple tree returns the Top Cycle. Similarly, the set of all co-winners for the Banks tree is the Banks set. Following Conitzer et al. [6] we phrase this in terms of parallel universes. Here the set of parallel universes is the set of permutations. Each parallel universe outputs a different (single) winner based upon a permutation of the alternatives. The parallel universe rule then takes the set of all of these. For a given binary tree ω we will write the parallel universe tournament function as ω_{PU}. Formally[3]:

$$\omega_{\mathsf{PU}}(T) = \{\ [\![\sigma(\omega)]\!](T)\ \mid\ \text{for all permutations } \sigma : X \to X\}$$

Similarly for a recursive tree function f we write the parallel universe tournament solution as f_{PU}.

Although Horan [10] gives necessary and sufficient conditions as to which tournament solutions are implementable in this manner, it is not generally noted that there may be multiple trees that implement the same tournament rules. In particular, the set of all co-winners for the iterative Condorcet tree is also the Top Cycle, i.e. $\mathsf{st}_{\mathsf{PU}} = \mathsf{it}_{\mathsf{PU}}$. However, there is no general bijection between permutations of the simple tree and permutations of the iterative Condorcet tree. Table 3 in the appendix shows the outcome for all permutations of our trees for the smallest non-trivial tournament, i.e. T of Example 1. Even in this small case involving only four alternatives we see that the number of permutations for which each alternative wins are different for the simple tree and the iterative Condorcet tree. Thus defining a probabilistic rule that determines the probability of selecting each alternative as the proportion of universes in which this alternative wins gives different results for the simple tree and for the iterative Condorcet tree. Let us use FR as a subscript for probabilistic rules based upon the proportion of universes in which alternatives win.

$$\omega_{\mathsf{FR}}(T)(c) = \frac{|\{\sigma \mid [\![\sigma(\omega)]\!](T) = c\}|}{|X|!}$$

We can also define a refinement of the parallel universe version by selecting those alternatives that win in the largest number of universes. We will describe the winners of this refinement as the *argmax* winners. Let us use AM as the subscript for rules that return the argmax winners.

$$\omega_{\mathsf{AM}}(T) = \operatorname*{argmax}_{c \in X} |\{\sigma \mid [\![\sigma(\omega)]\!](T) = c\}|$$

As for the parallel universe rules, we also have frequency and argmax versions of recursive families of trees f, respectively f_{FR} and f_{AM}.

All the parallel universe rules defined with respect to the trees above correspond to known rules, while the argmax winners are typically new refinements of these. To the best of our knowledge, the randomised rules we propose have not been studied theoretically as choice rules. We summarize the status of these rules in Table 1.

4 Properties and comparison of the rules

In this section we analyse the properties that our rules satisfy. The most straightforward results state that all rules of a certain type must satisfy a given property γ. Other results may be classed as "inheritance" results. These state (for instance) that if a given version of the rule (single tree, parallel universe, probabilistic or argmax) has property γ, this implies that a different version of the rule must also have property γ. We structure the following by considering in turn the parallel, probabilistic and argmax rules in terms of their general and inherited properties. In the subsection after these we apply these general results and determine which properties apply to the specific rules

[3]Here we abuse notation in applying the permutation σ directly to the tree and not the leaf alternatives.

Table 1: Summary of tree based rules.

	Parallel universe	Argmax	Randomised
Simple tree	Top Cycle [13]	new	[7]
Banks tree	Banks [13]	new	new
Fair tree	[12]	new	new
Two-leaf tree	Condorcet non-losers	Copeland	new
Iterative Condorcet tree	Top Cycle [1]	new	[1]

we have defined above. The final subsection deals with a slightly different issue: how much more effective the argmax rules are at selecting small sets of winners.

We must first define probabilistic versions of Condorcet consistency, monotonicity, composition consistency and stability. To each of these we prepend a P to indicate that it is the probabilistic version. (Some similar definitions are given in [3].) In the following $P(X)$ is the set of probability distributions over the outcomes X and $\mathsf{supp}(P(X))$ is the support of $P(X)$, i.e., the set of alternatives with a positive probability of winning. A randomised solution for a tournament will be denoted as $R(T)$ where $R : T \to P(X)$ is a function to a probability distribution over X, i.e., $R(T)(a)$ is the probability that alternative $a \in X$ wins.

We start with the notion of Condorcet consistency: if a tournament has a Condorcet winner, no other alternative should have a positive probability to be elected.

Definition 4.1 (P-condorcet consistency)**.** *If T has a Condorcet winner a, then $R(T)(a) = 1$.*

A weaker version would be to only require that the Condorcet winner, when it exists, has the largest probability of winning of all alternatives.

Definition 4.2 (Weak p-condorcet consistency)**.** *If T has a Condorcet winner a, then $R(T)(a) > R(T)(b)$ for all $b \neq a$.*

In a probabilistic setting, the simplest definition of monotonicity simply requires that if we reinforce a winner, her probability of winning cannot decrease.

Definition 4.3 (P-monotonicity)**.** *We say that R is p-monotonic if for any tournament T where bTa we have $R(T_{\langle a,b\rangle})(a) \geq R(T)(a)$.*

As with the deterministic versions, probabilistic composition consistency conditions concern changes to components of the tournament. Also as before, the weak version only requires a function with a domain of tournaments over a fixed set of alternatives, while the full version requires that we can calculate the outcomes for subtournaments.

Definition 4.4 (Weak p-composition consistency)**.** *Given a decomposable tournament $T = (T^*, T_{X_1}, \ldots, T_{X_k})$ and two alternatives $a, b \in X_i$ in some component, we require for any $c \notin X_i$, $R(T)(c) = R(T_{\langle a,b\rangle})(c)$.*

Note that this implies that changing the tournament within a component doesn't change the probability of selecting some alternative from the component, i.e. for a and b in component X_i we have $\sum_{d\in X_i} R(T)(d) = \sum_{d\in X_i} R(T_{\langle a,b\rangle})(d)$.

Definition 4.5 (P-composition consistency)**.** *If T can be decomposed into $(T^*, T_{X_1}, \ldots, T_{X_k})$ then for all $j \in \{1, \ldots, k\}$ and for all $x \in X_j$ $R(T)(x) = R(T_{X_j}) \cdot R(T^*)(j)$.*

For probabilistic stability, we first define a notion of S-equivalence.

Definition 4.6 (S-equivalence)**.** *Let P be a probability distribution over Y and P' a probability distribution over Z. Then P and P' are S-equivalent iff $\mathsf{supp}(P) = \mathsf{supp}(P')$ and for all $x \in \mathsf{supp}(P)$ we have $P(x) = P'(x)$.*

Note that when $Y = Z$ in the above, S-equivalence is effectively the identity.

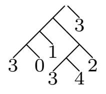
Definition 4.7 (P-stability)**.** *$R(T_Y)$ is S-equivalent to $R(T_Z)$ iff $R(T_{Y \cup Z})$ is S-equivalent to $R(T_Y)$*

4.1 Properties of parallel universe rules

In general parallel universe rules may not be monotonic. Laslier [13] provides a large counterexample.[4] We provide a smaller example with only 5 alternatives.

Example 3. *Consider the binary tree $\omega = 301(342)3$ and the tournament T with $4T3T2T1T0$ and iTj for all other $i < j$.*

$$T = \begin{tikzcd} 0 \\ 1 \arrow[u] \\ 2 \arrow[u] \\ 3 \arrow[u] \\ 4 \arrow[u] \end{tikzcd} \qquad T_{\langle 3,4 \rangle} = \begin{tikzcd} 0 \\ 1 \arrow[u] \\ 2 \arrow[u] \\ 3 \arrow[u] \\ 4 \end{tikzcd}$$

$301(342)3 =$ 

(Undrawn arrows go downwards.)

It can be verified that $\llbracket \omega \rrbracket(T) = 3$. In fact there are four permutations of the tree that select this alternative. Now reinforce alternative 3, obtaining $T_{\langle 3,4 \rangle}$. There are no permutations of ω that select 3 for the tournament $T_{\langle 3,4 \rangle}$ (verified by computer). That is to say, $3 \in \omega_{\mathsf{PU}}(T)$ but $3 \notin \omega_{\mathsf{PU}}(T_{\langle 3,4 \rangle})$.

However monotonicity is straightforwardly inherited from single universes to parallel universes.

Proposition 4.8. *If $\llbracket \omega \rrbracket$ is monotonic, then so is ω_{PU}.*

In particular this implies that $\mathsf{ft}_{\mathsf{PU}}$ is monotonic (the other particular parallel universe rules are well known to be monotonic). Weak composition consistency is not only inherited in the same manner, it further holds for all single universe and parallel universe rules based on binary trees.

Proposition 4.9. *For any ω, $\llbracket \omega \rrbracket$ and ω_{PU} are weakly composition consistent.*

For proof, see [13, 16].

In considering the remaining two properties, we need to shift attention from single binary trees to recursively defined families of trees, corresponding to the shift from tournament functions to tournament solutions. In general, for a function f defining a family of trees, both composition consistency and stability may or may not be satisfied by f_{PU}. The Banks tree provides an example of the first and a counterexample to the second. Conversely, the simple tree provides a counterexample to the first and an example of the second [4]. However, if the probabilistic version of a rule satisfies these properties, then so does the parallel universe version.

[4]More precisely, a family of counterexamples.

Proposition 4.10. *For a recursive family of trees f, if f_{FR} is p-composition consistent then f_{PU} is composition consistent.*

Proof. Suppose f_{FR} is p-composition consistent. Consider the support for the summary and componenets of any decomposed tournament. (I.e. note $\mathsf{supp}(f_{\mathsf{FR}}(T^*)) = f_{\mathsf{PU}}(T^*)$ and for each component with subscript i, $\mathsf{supp}(f_{\mathsf{FR}}(T_{X_i})) = f_{\mathsf{PU}}(T_{X_i})$.) □

Proposition 4.11. *For a tree function f, if f_{FR} is p-stable then f_{PU} is stable.*

Proof. As with p-composition consistency, consider the support. □

4.2 Properties of probabilistic rules

Similarly to parallel universe rules, monotonicity is straightforwardly inherited from single trees.

Proposition 4.12. *If $[\![\omega]\!]$ is monotonic, then so is ω_{FR}.*

Proposition 4.13. *For any non-singleton tree ω, ω_{FR} is weakly p-Condorcet consistent. If ω is also complete, ω_{FR} is p-Condorcet consistent.*

Proof. Supposing there are k different alternatives in the tree, a Condorcet winner will win in $k \cdot (m-1)!$ universes (in constructing the tree, there are k choices for the Condorcet winner within the tree, $m-1$ choices for the other alternatives). Any other alternative can only win in at most $k \cdot (m-k) \cdot (m-2)!$ universes (k choices for the alternative within the tree, $m-k$ choices for the Condorcet winner outside the tree, $m-2$ choices for the other alternatives). As the tree is non-singleton, $k > 1$, thus the Condorcet winner wins in more universes. □

Proposition 4.14. *For any tree ω, ω_{FR} is weak p-composition consistent.*

We have seen that composition consistency is inherited from probabilistic to parallel universe rules. However, the Banks tree in Table 3 provides a counterexample to inheritance in the opposite direction, as $\mathsf{bt}_{\mathsf{PU}}$ is composition consistent but $\mathsf{bt}_{\mathsf{FR}}$ is not.

4.3 Properties of argmax rules

Argmax rules refine their parallel universe versions by definition. Thus the following is direct in the case of complete trees.

Proposition 4.15. *Any argmax binary tree rule is Condorcet consistent.*

Proof. This follows easily from Proposition 4.13. □

Weak composition consistency (and thus also composition consistency) is always violated by argmax rules.

Proposition 4.16. *Any argmax rule violates weak composition consistency.*

Proof. Consider the tournament $T = (T^*, T_1, 2, 3)$ and $T' = (T^*, T_2, 2, 3)$ such that T^* and T_1 are cyclic tournaments with 3 alternatives, T_2 is a transitive tournament with 3 alternatives, and 2 and 3 are tournaments with single alternatives. Suppose for a contradiction that there is a tree ω such that ω_{AM} is composition consistent. This must make all five candidates winners in T, by composition consistency and neutrality. As there are $5! = 120$ permutations of the five candidates, each candidate in T must win for 24 permutations. By weak p-Condorcet consistency the candidates in 2 and 3 must also win for 24 permutations, but the unique winner of T_2 wins for the remaining 72 permutations, thus is the counting rule winner, violating weak composition consistency. □

Table 2: Known properties (deterministic/randomised) for trees.

	Monotonicity			Composition Consistency			Stability		
	PU	FR	AM	PU	FR	AM	PU	FR	AM
Simple tree	✔	✔	?	✘	✘	✘	✔	✔	?
Iterative Condorcet tree	✔	✔	?	✘	✘	✘	✔	✔	?
Banks tree	✔	✔	?	✔	✘	✘	✘	✘	?
Balanced tree	✔	✔	?	✘	✘	✘	✘	✘	?
Two-leaf tree	✔	✔	✔	✘	✘	✘	✘	✘	✘

Monotonicity for argmax is an interesting open problem. We note that it is distinct from monotonicity or p-monotonicity. The only result we have here is for the two-leaf tree, which forms a known rule (Copeland) under argmax. The possibility of failure of monotonicity for arbitrary trees under parallel universes strongly suggests that this will also be the case here, but the more interesting case concerns rules that are monotonic in the single and parallel universe cases: is this then inherited by argmax?

4.4 Properties of specific trees

With respect to the probabilistic rules and monotonicity: the simple, balanced and two-leaf trees are non-repetitive, thus monotonic. We prove that the Banks and iterative Condorcet trees are also monotonic below. By the inheritance results above we also have monotonicity for both parallel universe and probabilistic versions of all these rules.

Proposition 4.17. *The Banks tree is monotonic.*

Proof. This can be seen to be monotonic by considering the following high level description of how the tree reduction proceeds here. First, we suppose that the rightmost alternative in the Banks tree is the preliminary winner. We then successively examine the other alternatives (as arguments in the recursive definition from right to left), potentially setting them as new preliminary winners. For a alternative to become the new preliminary winner, it must defeat every member of the set of previous preliminary winners. After all the alternatives have been tested, we select the current preliminary winner. Clearly, if an alternative was selected then it defeated all previous preliminary winners: changing only this alternative so that it defeats more alternatives will not change the fact that it is selected. □

Proposition 4.18. *The iterative Condorcet tree is monotonic.*

Proof. Consider the process definition that this tree emulates: elminate alternatives according to some fixed ordering until there is a Condorcet winner. If the alternative was the first Condorcet winner before, then it is not possible that another alternative becomes a Condorcet winner before this alternative during the same elimination process. □

We now give a short discussion on the balanced tree: though the simple and Banks trees correspond to known tournament solutions and have intuitive descriptions as processes, the balanced tree is somewhat more difficult to understand. By Table 3 it is easy to see that it is neither weakly composition consistent nor stable. The parallel universe version of this rule is sometimes called the cup rule in the literature, however it is not typically considered in relation to other tournament solutions.

This may be because it is neither a superset nor a subset of the uncovered set, which perhaps marks it as an undesirable tournament solution. It may nonetheless be interesting from a randomised or argmax perspective.

Inheritance of stability from probabilistic to parallel universe rules settles (by contraposition) questions of p-stability for any non-stable parallel universe rules. The following result concerning the two remaining stable parallel universe rules leaves open the probability that stability is inherited from parallel universe to probabilistic rules.

Proposition 4.19. $\mathsf{st}_{\mathsf{FR}}$ *and* $\mathsf{it}_{\mathsf{FR}}$ *satisfy p-stability.*

Proof. (Only if) If $R(T_X)$ and $R(T_Y)$ are S-equivalent for either of these rules then the Top Cycle of T_X is the same as the Top Cycle of T_Y, and both are the same as the Top Cycle of $T_{X\cup Y}$. Consider any permutation that produces alternative c as a winner for T_X. Adding the alternatives in $Y\backslash X$ to this permutation in any order will not change the winner under the expanded tree. Thus the proportion of permutations for which c is a winner remains the same.

(If) Similarly, as the alternatives in $Y\backslash X$ have no effect on the permutation, there must be the same proportions of permutations producing each alternative winner after these are removed. □

Collecting together all these results, we summarise the known properties for our specific tress in Table 2. As we have already noted, it is not clear whether or not we have general inheritance of monotonicity for the argmax rules. Nor have we proven or disproven monotonicity in specific cases. However, we conjecture that all the trees we have considered are monotonic under argmax.

4.5 Success of argmax versions as refinements

We may question how reasonable the argmax rules are. Aside from the (open) question of monotonicity, we are interested in how effective they are at actually refining the set of winners. We have tested this on some example tournaments. The outcome of all of our rules only concern alternatives in the Top Cycle: any Condorcet losers will not affect the outcome of the vote. Thus we restrict attention to what are called non-reducible tournaments, those in which the whole tournament is returned by the Top Cycle. Moon [15] provides a list of all non-isomorphic small tournaments, from which we see that there are only 34 non-reducible tournaments of size 6. We applied our rules to all of these, and compared them with the Markov solution concept, generally considered among the most decisive tournament solutions. The specific results are found in Table 4 in the appendix. From this table it can be verified that all these rules are distinct. We can also see that the Banks set contains three alternatives 14 times, four alternatives 8 times, five alternatives 9 times and six alternatives 3 times. In contrast $\mathsf{bt}_{\mathsf{AM}}$ outputs a single winner 32 times, two winners 1 time and three winners 2 times. Both $\mathsf{st}_{\mathsf{AM}}$ and $\mathsf{it}_{\mathsf{AM}}$ get similar (though distinct) results. Copeland, often considered a fairly decisive solution concept, which is here equivalent to $\mathsf{tt}_{\mathsf{AM}}$, outputs a single winner 18 times, two winners 7 times, three winners 5 times and four winners 4 times. Thus it appears that the argmax rules are significantly more decisive than the full parallel universe versions.

5 Conclusion

In this paper, we propose a general principle for constructing randomised voting rules (or social decision schemes) and neutral refinements of voting rules. We use the idea that, given a voting rule, some object that we call a universe can ensure that the voting rule is resolute. One example is to consider that a universe is defined by a tie-breaking rule. In this paper, we have considered that a universe is defined by a particular assignment of alternatives to a voting tree. There are three types of rules that are naturally definable with respect to a set of universes: parallel rules that output all winners for all possible universes, probabilistic (frequency) rules that randomly pick a universe and output the winner from that universe, and argmax rules that output the winners of the most universes.

We study whether properties can be inherited between these different versions of rules. We plan to complete the study of properties that can be inherited in the context of tournaments. For voting rules, our principle may be interesting for voting rules such as Instant-runoff voting (IRV) and Ranked Pairs: for former, there may be ties to be broken between the alternatives with the smallest number of votes. We can use our principle to define a new neutral refinement of IRV and a randomised rule based on IRV. It would be interesting to study the properties of such a rule.

We have noted that the argmax rules are attractive in that they output smaller sets of winners than their parallel version conterparts. However, the real test of their attractiveness hinges upon whether or not they are monotonic, a property we have not been able to prove or disprove.

We have not yet considered computational issues, which will probably be a heavy burden. We expect that some winner determination problems will be in the class #-P and we leave this study as future work. However, we also plan to investigate to what extent Monte Carlo methods can estimate winners. If computing the winner is easy in a given universe, we may require not too many samples to estimate the winner of the argmax rule or to have a good approximation of the randomised rule.

References

[1] Alon Altman and Robert Kleinberg. Nonmanipulable randomized tournament selections. In *AAAI*, 2010.

[2] Jeffrey S Banks. Sophisticated voting outcomes and agenda control. *Social Choice and Welfare*, 1(4):295–306, 1985.

[3] Florian Brandl, Felix Brant, and Hans G Seedig. Consistent probabilistic social choice (forthcoming). *Econometrica: Journal of the Econometric Society*, 2016.

[4] Felix Brandt, Markus Brill, and Paul Harrenstein. Tournament solutions. In *Handbook of Computational Social Choice*, chapter 3. Cambridge University Press, 2016.

[5] Markus Brill and Felix Fischer. The price of neutrality for the ranked pairs method. In *Proceedings of AAAI-12*, pages 1299–1305, 2012.

[6] Vincent Conitzer, Matthew Rognlie, and Lirong Xia. Preference functions that score rankings and maximum likelihood estimation. In *Proceedings of IJCAI-09*, pages 109–115, 2009.

[7] Felix Fischer, Ariel D Procaccia, and Alex Samorodnitsky. A new perspective on implementation by voting trees. In *Proceedings of the 10th ACM conference on Electronic commerce*, pages 31–40. ACM, 2009.

[8] Rupert Freeman, Markus Brill, and Vincent Conitzer. General tiebreaking schemes for computational social choice. In *Proceedings of AAMAS-15*, 2015.

[9] Sean Horan. Implementation by agenda voting. *COMSOC 2012*, pages 239–250, 2012.

[10] Sean Horan. Implementation of majority voting rules. *preprint*, 2013.

[11] Olivier Hudry. A note on “Banks winners in tournaments are difficult to recognize” by G. J. Woeginger. *Social Choice and Welfare*, 23(1):113–114, 2004.

[12] Jérôme Lang, Maria Silvia Pini, Francesca Rossi, Kristen Brent Venable, and Toby Walsh. Winner determination in sequential majority voting. In *IJCAI'07*, volume 7, pages 1372–1377, 2007.

[13] Jean-François Laslier. *Tournament Solutions and Majority Voting*. Springer, 1997.

[14] Nicholas R Miller. Graph-theoretical approaches to the theory of voting. *American Journal of Political Science*, pages 769–803, 1977.

[15] John W Moon. *Topics on Tournaments in Graph Theory*. Holt, Rinehart and Winston, 1968.

[16] Hervé Moulin. Choosing from a tournament. *Social Choice and Welfare*, 3(4):271–291, 1986.

[17] Sanjay Srivastava and Michael A Trick. Sophisticated voting rules: The case of two tournaments. *Social Choice and Welfare*, 13(3):275–289, 1996.

[18] T Nicolaus Tideman. Independence of clones as a criterion for voting rules. *Social Choice and Welfare*, 4(3):185–206, 1987.

[19] Michael A Trick. Small binary voting trees. 2006.

[20] Gerhard. J. Woeginger. Banks winners in tournaments are difficult to recognize. *Social Choice and Welfare*, 20(3):523–528, 2003.

Appendix

Table 3: Alternatives selected by each permutation σ of a recursive tree g, applied to the tournament T of Example 1. The values in the top half of the table are $[\![g(\sigma)]\!](T)$.

σ	$g = \mathsf{st}$	$g = \mathsf{bt}$	$g = \mathsf{ft}$	$g = \mathsf{tt}$	$g = \mathsf{it}$
1234	1	3	3	1	3
2134	2	3	3	1	3
3214	3	3	4	2	4
2314	2	3	4	2	3
3124	3	3	3	3	4
1324	3	3	3	3	4
4321	3	3	3	3	1
3421	3	3	3	3	4
3241	3	3	4	2	4
4231	4	3	3	4	3
2431	2	3	3	4	3
2341	2	3	4	2	3
4123	4	2	4	4	2
1423	4	2	4	4	2
1243	1	2	3	1	3
4213	4	2	3	4	3
2413	2	2	3	4	3
2143	2	2	3	1	3
4132	4	4	4	4	2
1432	4	4	4	4	2
1342	3	4	3	3	4
4312	3	4	3	3	1
3412	3	4	3	3	4
3142	3	4	3	3	4
Number of permutations for which a is selected:					
$a = 1$	2	0	0	4	2
$a = 2$	6	6	0	4	4
$a = 3$	10	12	16	8	10
$a = 4$	6	6	8	8	8

Table 4: Selected alternatives for various rules for all non-reducible tournaments of size 6. The alternatives are labelled from 0 to 5.

$\mathsf{st}_{\mathsf{AM}}$	$\mathsf{bt}_{\mathsf{AM}}$	$\mathsf{ft}_{\mathsf{AM}}$	$\mathsf{tt}_{\mathsf{AM}}$	$\mathsf{it}_{\mathsf{AM}}$	markov
0	0	0	0,1	0	0
0	0	0	0,1	0	0
0	0	0	0,1	0	0
0	0	0	0,1	0	0
0	1	0	0	0	0
1	1	1	1	1	1
0	0	0	0	0	0
0	0	0	0,1	0	0
0	0	0	0,1	0	0
0	0	0	0,1	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
2	2	2	0	0	0
0	0,1	0	0	0	0
0	0	0	0	0	0
0	4	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0,1,2,3	0	0
4	4	0	0,1,2,4	0	4,0
0	0	0	0,1,2,3	0	0
1	1	1,2	1,2,3,4	1	1
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
2	2	2	2,3,4	2	2
1	1	1,3,4	1,3,4	1,3	1
4	4	4	1,4,5	4	4
3,4,5	3,4,5	3,4,5	3,4,5	3,4,5	5,4,3
3,4,5	3,4,5	3,4,5	3,4,5	3,4,5	5,4,3

Justin Kruger
LAMSADE
Université Paris-Dauphine
PSL Research University
Paris, France
Email: `justin.g.kruger@gmail.com`

Stéphane Airiau
LAMSADE
Université Paris-Dauphine
PSL Research University
Paris, France
Email: `stephane.airiau@dauphine.fr`