A Study of Proxies for Shapley Allocations of Transport Costs*

Haris Aziz
NICTA and UNSW
Sydney, Australia

Casey Cahan
University of Auckland
Auckland, New Zealand

Charles Gretton
NICTA and ANU
Canberra, Australia

Phillip Kilby
NICTA and ANU
Canberra, Australia

Nicholas Mattei
NICTA and UNSW
Sydney, Australia

Toby Walsh
NICTA and UNSW
Sydney, Australia

Abstract

We propose and evaluate a number of solutions to the problem of calculating the cost to serve each location in a single-vehicle transport setting. Such cost to serve analysis has application both strategically and operationally in transportation. The problem is formally given by the *traveling salesperson game* (TSG), a cooperative total utility game in which agents correspond to locations in a travelling salesperson problem (TSP). The cost to serve a location is an allocated portion of the cost of an optimal tour. The *Shapley value* is one of the most important normative division schemes in cooperative games, giving a principled and fair allocation both for the TSG and more generally. We consider a number of direct and sampling-based procedures for calculating the Shapley value, and present the first proof that approximating the Shapley value of the TSG within a constant factor is NP-hard. Treating the Shapley value as an ideal baseline allocation, we then develop six proxies for that value which are relatively easy to compute. We perform an experimental evaluation using Synthetic Euclidean games as well as games derived from real-world tours calculated for fast-moving consumer goods scenarios. Our experiments show that several computationally tractable allocation techniques correspond to good proxies for the Shapley value.

Introduction

We study transport scenarios where deliveries of consumer goods are made from a depot to locations on a road network. At each location there is a customer, e.g. a vending machine or shop, that has requested some goods, e.g. milk, bread, or soda. The vendor who plans and implements deliveries is faced with two vexing problems. First, the familiar combinatorial problem of routing and scheduling vehicles to deliver goods cost effectively. Many varieties of this first problem exist (Golden, Raghavan, and Wasil 2008), and for our proposes we shall refer to it as the *vehicle routing problem* (VRP). We begin our investigation supposing that VRP has been solved heuristically, and therefore after the assignment of locations to routes has been made.

The second vexing problem is determining how to evaluate the *cost to serve* each location. Specifically, the vendor must decide how to apportion the costs of transportation to each location in an equitable and economically efficient manner. The results of cost to serve analysis have a variety of important applications. Using the allocation directly the vendor can of course charge locations their allocated portion of the transportation costs. More realistically, vendors use the cost allocations when (re-)negotiating contracts with customers. Supply chain managers may also reference a cost allocation when deciding whether or not to continue trade with a particular location. Finally, provided market conditions are favourable, sales managers can be instructed to acquire new customers in territories where existing cost allocations are relatively high in order to share the cost of delivery among more locations.

Addressing the second vexing problem, this paper stems from our work with a fast-moving consumer goods company that operates nationally both in Australia and New Zealand. The company serves nearly 20,000 locations weekly using a fleet of 600 vehicles. Our industry partner is under increasing economic pressure to realise productivity improvements through optimisation of their logistical operations. A key aspect of that endeavour is to understand the contribution of each location to the overall cost of distribution. In this study we focus at the individual route level for a single truck, where we apportion the costs of the deliveries on that route to the constituent locations. We formalise this setting as a *traveling salesperson game* (TSG) (Potters, Curiel, and Tijs 1992), where the cost to serve all locations is given by the solution to an underlying *traveling salesperson problem* (TSP). Formalised as a game, we can use principled solution concepts from cooperative game theory, notably the Shapley value (Shapley 1953), in order to allocate costs to locations in a fair and economically efficient manner.

Calculating the Shapley value of a game is a notoriously hard problem (Chalkiadakis, Elkind, and Wooldridge 2011). A direct calculation for a TSG requires the optimal solution to exponentially many distinct instances of the TSP. Though sampling can provide a rough approximation, we cannot hope to achieve better. We prove that there is no polynomial-time α-approximation of the Shapley value for any constant $\alpha \geq 1$ unless P = NP.

To circumscribe these computationally difficulties, this work explores six proxies[1] for the Shapley value. Our prox-

*An extended version of this work, including detailed proofs of theorems can be found at http://arxiv.org/abs/1408.4901.

[1]We use the word *proxy* instead of *approximation* to ease dis-

ies offer tractable alternatives to the Shapley value, and in some cases appeal to other allocation concepts from cooperative game theory (Peleg and Sudhölter 2007; Curiel 2008). Two of our proxies appeal to the well-known *Held-Karp* and *Christofides* TSP heuristics, respectively.

We report a detailed experimental comparison of proxies using a large corpus of Synthetic Euclidean games, and problems derived from real-world tours calculated for fast-moving consumer goods businesses in the cities of Auckland (New Zealand), Canberra, and Sydney (Australia). We highlight three computationally tractable proxies that give good approximations of the Shapley value in practice. Our evaluation also considers the ranking of locations—least to most costly—induced by the Shapley and proxy values. Ranking is relevant when, for example, we are just interested in identifying the most costly locations to serve. We again find that three of our proxies provide good ranking accuracy taking the rank induced by the Shapley value as the target.

Preliminaries

We use the framework of cooperative game theory to gain a deeper understanding of our delivery and cost allocation problems (Peleg and Sudhölter 2007; Chalkiadakis, Elkind, and Wooldridge 2011). In cooperative game theory, a game is a pair (N,c). N is the set of agents and the second term $c : 2^N \to \mathbb{R}$ is the *characteristic function*. Taking $S \subseteq N$, $c(S)$ is the cost of subset S. A *cost allocation* is a vector $x = (x_0,\ldots,x_n)$ denoting that cost x_i is allocated to agent $i \in N$. We restrict our attention to economically *efficient* cost allocations, which are allocations satisfying $\sum_{i\in N} x_i = c(N)$.

For any cooperative game (N,c), a *solution concept* ϕ assigns to each agent $i \in N$ the cost $\phi_i(N,c)$. There may be more than one allocation satisfying the properties of a particular solution concept, thus ϕ is not necessarily single-valued, and might give a set of cost allocations (Peleg and Sudhölter 2007). A minimal requirement of a solution concept is *anonymity*, meaning that the cost allocation must not depend on the identities of locations. Prominent solution concepts include the *core*, *least core*, and the *Shapley value*. For $\varepsilon \geq 0$, we say that cost allocation ϕ is in the (multiplicative) ε-*core* if $\sum_{i\in S} \phi_i \leq (1+\varepsilon)c(S)$ for all $S \subseteq N$ (Faigle and Kern 1993). The 0-core is referred to simply as the *core*. Both the core and ε-core can be empty. The ε-core which is non-empty for the smallest possible ε is called the least core. This particular ε is referred to as the *least core value*.[2]

Our work focuses on the single-valued solution concept called the *Shapley value* (Shapley 1953). Writing $SV_i(N,c)$ for the Shapley value of agent i, formally we have: $SV_i(N,c) = \sum_{S\subset N\setminus\{i\}} \frac{|S|!(|N|-|S|-1)!}{|N|!}(c(S\cup\{i\})-c(S))$. In other words, the Shapley value divides costs based on the marginal cost contributions of agents

In the *traveling salesperson problem (TSP)* a salesperson must visit a set of locations $N = \{1,\ldots,n\}\cup\{0\}$ starting and ending at a special *depot* location 0. For $i,j \in N\cup\{0\}$ $i \neq j$, d_{ij} is the strictly positive distance traversed when traveling from location i to j. Here, $d_{ij} = \infty$ if traveling directly from i to j is impossible. Taking distinct $i,j,k \in N\cup\{0\}$, the problem is *symmetric* if and only if $d_{ij} = d_{ji}$ for all $i,j \in N\cup\{0\}$. It satisfies the *triangle inequality* if and only if $d_{ij} + d_{jk} \geq d_{ik}$ (Garey and Johnson 1979).

A TSP is *Euclidean* when each location is given by coordinates in a (two dimensional) Euclidean space; therefore d_{ij} is the Euclidean distance between i and j. A Euclidean TSP is both symmetric and satisfies the triangle inequality.

A *tour* is given by a finite sequence of locations that starts and ends at the depot 0. The *length* of a tour is the sum of distances between consecutive locations. For example, the length of $[0,1,2,0]$ is $d_{01}+d_{12}+d_{20}$. An optimal solution to a TSP is a minimum length tour that visits every location. It is NP-hard to find an optimal tour, and generally there is no α-approximation for any α unless P = NP. An α-approximation for a given optimisation problem is an algorithm that runs on an instance x and returns a feasible solution $F(x)$ which has cost $c(F(x))$ related to the optimal solution $OPT(x)$ by the following relation (Papadimitriou 1994): $\frac{|c(F(x))-c(OPT(x))|}{\max\{c(OPT(x)),c(F(x))\}} \leq \alpha$. Informally, α is a bound on the relative error of an approximation function. When $\forall i,j$ d_{ij} are finite, the triangle inequality and symmetry hold, then polynomial-time approximations exist (Held and Karp 1962; Christofides 1976).

Given a TSP, the corresponding *traveling salesperson game* (TSG) is a pair (N,c). N is the set of agents which corresponds to the set of locations.[3] The second term $c : 2^N \to \mathbb{R}$ is the characteristic function. Taking $S \subseteq N$, $c(S)$ is the length of the shortest tour of all the locations in S. A *cost allocation* is a vector $x = (x_1,\ldots,x_n)$ denoting that cost x_i is allocated to location $i \in N$. For the special depot location, we shall always take $x_0 = 0$ (Potters et al. 1992).

Some Properties of the Shapley Value

The Shapley value has many attractive properties when used as a cost allocation scheme by a vendor. For example, whereas the 0-core can be empty, and therefore not yield any allocation at all (Tamir 1989), the Shapley value always exists in the TSG setting. The Shapley value is also, for general games, the unique assignment of costs that satisfies three important properties: (1) *anonymity*, the cost allocated to a particular location is dependent only on the impact it has to the total cost; (2) *efficiency*, the entire cost of serving all N locations is allocated; and (3) *strong monotonicity*. The latter states that if the total cost of a coalition is reduced, then the allocation to all locations participating in that coalition is either reduced or not increased (Young 1985). Formally, the marginal contribution from player i to the total cost of coalition S is $c^i(S) = c(S) - c(S\setminus\{i\})$if $i \in S$ and $c(S\cup\{i\}) - c(S)$if $i \notin S$. Strong monotonicity can be stated as: $\forall S : c^i(S) \geq c'^i(S) \Longrightarrow \phi_i(N,c) \geq \phi_i(N,c')$. Due to these

cussion and, technically, many of these measures are stand-ins for the Shapley value, not approximations of it.

[2]The 0-core of the transport game we focus on in this work can be empty. However, if the game is convex, the Shapley value lies in the core (Tamir 1989).

[3]From here on we focus on a restriction of general games to delivery games (TSGs) and therefore we use *location* instead of *agent* for ease of exposition.

and other derivative axiomatic properties, the Shapley value has been termed "the most important normative payoff division scheme" in cooperative game theory (Winter 2002).

Another important property of the Shapley value is that it would allocate any fixed costs incurred when serving a location to that location alone. If we treat a variant of the TSG where some locations have an associated fixed cost in addition to their transportation costs— e.g. parking and loading fees —then the Shapley value will allocate those fixed costs to the associated locations. Formally, given a fixed cost $f(i)$ of serving location i, $f(i)$ does not need to be removed before computing the Shapley value, as follows. Suppose c is the characteristic function of the TSG defined above, and c' satisfies the identity $c'(S) = c(S) + \sum_{i \in S} f(i)$.

Proposition 1 $SV_i(N,c') = SV_i(N,c) + f(i)$.

We also have that by charging locations according to the Shapley value, we can expect to incentivize them to *recruit* new customers in their vicinity. Locations recruiting for a vendor can reasonably expect to lower the transportation costs they are allocated. In detail, consider a vendor trading with locations $N = \{1..|N|\}$. From the vendors perspective, adding a new location, $|N|+1$, to an existing delivery route is clearly a good idea if the revenue generated by delivering to that location is greater than the marginal cost $c(N \cup \{|N|+1\}) - c(N)$ of the new delivery. Because existing locations in the vicinity of $|N|+1$ are already paying for deliveries, charging at the threshold $c(N \cup \{|N|+1\}) - c(N)$ however will typically be unfair. In that case existing customers would likely be subsidizing new customers, and therefore disincentivize to find new business for the vendor. The Shapley value mitigates this, and can be expected to provide recruitment incentives. Making this discussion more concrete, suppose the game is a Euclidean scenario with $N = \{x\}$ a single agent at distance 100 from the depot and the new agent y is at distance 5 from x. The transportation cost of serving $\{x,y\}$ can be as high as 210. Clearly, charging the new agent $c(\{x,y\}) - c(\{x\}) = 10$ while x continues to pay around 200 is unfair. However, if the vendor allocates costs according to the Shapley value, the existing customer's costs *decrease* when adding the new agent.

Related to the above discussion, if the characteristic function is concave then the Shapley value lies in the non-empty 0-core. Formally, concavity is satisfied if for all $S \subseteq N \setminus \{i\}$: $c(S \cup \{i\} \cup \{|N|+1\}) - c(S \cup \{|N|+1\}) < c(S \cup \{i\}) - c(S)$. Charging customers according to core values actually guarantees that they are incentivized to recruit. Specifically, for all $i \in N : SV_i(N \cup \{|N|+1\}, c) < SV_i(N,c)$. In other words, the Shapley allocation of costs to existing locations decreases when a new customer $|N|+1$ is added. Unfortunately general TSGs do not necessarily have concave characteristic functions. However, concavity in expectation is all that is required for existing locations to realise savings. In practice there are synergies, and incentives for further recruitment on routes where we charge according to the Shapley value. In our empirical data, even when the game is not concave we frequently observe such incentives given a Shapley allocation. And compared to charging customers according to their marginal contribution to costs, we do not explicitly disincentivize recruitment. Summarizing, if an agent knows that all locations are charged according to the Shapley value, they can typically expect incentives to recruit new locations in their vicinity.

Computing the Shapley Value

Our focus now shifts to calculation of the Shapley value. We require an accurate baseline in order to experimentally evaluate the proxies we later develop for the Shapley value of the TSG. A direct calculation for a TSG requires the optimal solution to exponentially many distinct instances of the TSP. We prove that there is no polynomial-time α-approximation of the Shapley value for any constant $\alpha \geq 1$ unless P = NP.

Theorem 2 *There is no polynomial-time α-approximation of the Shapley value of the location in a TSG for constant $\alpha \geq 1$ unless* P = NP.

Proof. Let $G(N,E)$ be a graph with nodes N and edges E. If an α-approximation exists we can use it to solve the NP-complete Hamiltonian cycle problem on G. First, from G construct a complete weighted and undirected graph $G'(N,E')$, where (i,j) has weight 1 if (i,j) is in the transitive closure of E, and otherwise has weight $n!\alpha$. If there is a Hamiltonian cycle in G then the Shapley value of any $i \in N$ in the TSG posed by G' is at most 1. Suppose there is no Hamiltonian cycle in G. We show there exists a permutation π of N that induces a large Shapley value for any node j as follows: repeatedly add a node from $N \setminus j$ to π so that there remains a Hamiltonian cycle amongst elements in π; when there is no such node then add j. The marginal cost of adding j to π is at least $n!\alpha$. The Shapley value of j is the average cost of adding it to a coalition $S \subseteq N \setminus j$, therefore its Shapley value is at least α. Even though edge weights in G' are large, we can represent G' compactly in $O(\log(n) + n^2 \log(\alpha))$ space. An α-approximation on G' for j decides the existence of the Hamiltonian cycle in G. ❑

Using the state-of-the-art TSP solver Concorde (Applegate et al. 2007) in a direct calculation of the Shapley value, we find it impractical to compute the exact Shapley value for instances of the TSG larger than about 15 locations. Any direct calculation method requires an exponential number of characteristic function computations, each requiring we solved an NP-hard problem.

To calculate the Shapley value of a TSG we employ the Type-0 method suggested by Mann and Shapley 1960, called *ApproShapley* by Castro, Gómez, and Tejada. The pseudocode is given in Algorithm 1. Writing $\pi(N)$ for the set of $|N|!$ permutation orders of locations N, taking $\Pi \in \pi(N)$ we write Π_i for the subset of N which precede location i in Π. An alternative formulation of the Shapley value can be characterised in terms of $\pi(N)$, by noting that value equates with marginal cost of each location when we construct coalitions in all possible ways:$SV_i(N,c) = \frac{1}{|N|!} \sum_{\Pi \in \pi(N)} (c(\Pi_i \cup \{i\}) - c(\Pi_i))$. For each sampled permutation, *ApproShapley* evaluates the characteristic function for each $i \leq |N|$ computing the length of an optimal tour for the set of locations in the i-sized prefix. By construction, the cost allocation produced

by *ApproShapley* is economically efficient. As a small but important optimization, in our work we cache the result of each evaluation of the characteristic function to avoid solving the same TSP twice.

Bachrach et al. have previously examined Type-0 sampling in *simple games*—i.e. cost of a coalition is either 0 or 1—deriving bounds that are *probably approximately correct*. In other words, the actual Shapley value lies within a given error range with high probability (Bachrach et al. 2010). Continuing in this line of work, Maleki et al. show that if the range or variance of the marginal contribution of the players is known ahead of time, then more focused (termed *stratified*) sampling techniques may be able to decrease the number of samples required to achieve a given error bound (Maleki et al. 2013). Other methods of approximating the Shapley value, specifically for weighted voting games, have appeared in the literature including those based on multi-linear extensions (Leech 2003; Owen 1972) and focused random sampling (Fatima et al. 2008; 2007).

Algorithm 1 ApproShapley

Input: $N = \{1, \ldots, n\}$ locations with cost $c(S)$ to serve a subset $S \subseteq N$ and m number of iterations.
Output: SV_i for all $i \in |N|$

$SV \leftarrow []$
for $i \leftarrow 1$ **to** $|N|$ **do**
  $SV_i \leftarrow 0$
end for
$SampleNumber \leftarrow 1$
for $SampleNumber \leftarrow 1$ **to** m **do**
  Randomly select a permutation of the locations *Perm* from $\pi(N)$
  $S \leftarrow \emptyset$
  for $i \leftarrow 1$ **to** $|N|$ **do**
    $S \leftarrow S \cup \{Perm_i\}$
    $SV_{Perm_i} \leftarrow SV_{Perm_i} + (c(S) - c(S \setminus \{Perm_i\}))$
  end for
end for
$TotalValue \leftarrow \sum_{i \in N} SV_i$
for $i \leftarrow 1$ **to** $|N|$ **do**
  $SV_i \leftarrow SV_i * (c(N)/TotalValue)$
end for
return SV

Proxies for the Shapley Value

The use of *ApproShapley* requires that we solve an NP-hard problem each time we evaluate the characteristic function. This is feasible for small TSG instances with less than a dozen locations, however it does create an unacceptable computational burden in larger, realistically sized games. We now describe a variety of proxies for the Shapley value that require much less computation in practice.

For the purposes of the discussion below we assume that an optimal tour for the underlying TSP is given. Not all our proxies yield economically efficient allocations of the cost of the optimal tour. For that reason, we define proxies in terms of the induced *fractional* allocation of the cost of the optimal tour. Later, we shall compare these fractional allocations to that induced by computing the fractional Shapley value, formally $\phi_i^{\text{SV}} = SV_i / \sum_{j \in n} SV_j$. This formulation based on fractional allocations allows us to compare the cost allocations from all the proxies on equal footing, in a way that would be used in operational contexts such as transport settings. This formulation also enables us to efficiently—i.e. in the game theoretic sense—allocate the cost of the optimal route only having to solve the NP-hard TSP once.

Depot Distance (ϕ^{DEPOT})

The distance from the depot — i.e. d_{i0} for location i — is our most straightforward proxy. We allocate cost to location i proportional to d_{i0}. The fraction allocation to location i is $\phi_i^{\text{DEPOT}} = \frac{d_{i0}}{\sum_{i=1}^{n} d_{i0}}$. For this proxy, a location that is twice as distant from the depot as another has to pay twice the cost.

Shortcut Distance (ϕ^{SHORT})

Another proxy that is straightforward to calculate and which has been used in commercial routing software is the *shortcut distance*. This is the marginal cost savings of skipping a location when traversing a given optimal tour. With no loss of generality, suppose the optimal tour visits the locations according to the sequence $[0, 1, 2, \ldots]$. Formally, $\text{SHORT}_i = d_{i-1,i} + d_{i,i+1} - d_{i-1,i+1}$, where locations 0 and $n+1$ are the depot, and d_{ij} is the cost of travel from location i to j. The fractional allocation given by the shortcut distance is then $\phi_i^{\text{SHORT}} = \frac{\text{SHORT}_i}{\sum_{j \in N} \text{SHORT}_j}$.

Re-routed Margin (ϕ^{REROUTE})

For a location $i \in N$, REROUTE_i is defined as $c(N) - c(N \setminus i))$. The allocation to a player can be computed with at most two calls to an optimal TSP solver. The fractional allocation is $\phi_i^{\text{REROUTE}} = \frac{(c(N) - c(N \setminus i))}{\sum_{j=N} (c(N) - c(N \setminus j))}$.

Christofides Approximation (ϕ^{CHRIS})

A more sophisticated proxy is obtained if we use a heuristic when performing characteristic function evaluations in *ApproShapley*, rather than solving the individual induced TSPs optimally. For this proxy we use sampling to estimate the Shapley value and we use an approximation algorithm to estimate the underlying TSP cost. To approximate the underlying TSP characteristic function, the Christofides heuristic (Christofides 1976), an $O(N^3)$ time procedure is used. To obtain a fractional quantity ϕ_i^{CHRIS}, we divide the allocation to location i by the sum total of allocated costs. Assuming a symmetric distance matrix satisfying the triangle inequality, the Christofides heuristic is guaranteed to yield a tour that is within $3/2$ the length of the optimal tour.

Nested Moat-Packing (ϕ^{MOAT})

A cost allocation method based on a nested moat-packing was first introduced by Faigle et al. 1998. This allocation is obtained by apportioning a grand-coalition cost equal to the value of the Held-Karp (Held and Karp 1962) relaxation of the underlying TSP, multiplied by a constant factor.

The proposed allocation is calculated by surrounding locations using a set of geometrically nested reagions called

moats. To obtain a cost allocation, moats are arranged so that for a vehicle to visit the set of locations in the underlying TSP, that vehicle must traverse the width of each moat at least twice. Choosing moats in order to to maximise the sum of their widths, the distance traversing all chosen moats twice corresponds to the value of the Held-Karp lower bound. One obtains an ε-core value by allocating each moat width twice to locations outside the moat, and then scaling those allocations, here by the constant factor 1.5, ensuring the sum of allocated costs exceeds the optimal tour length.

The above ideas is expressed mathematically below in the constraints and optimisation criterion in Equation 1. Formally, the moat width, w_S, for a set of locations $S \subseteq N$ is calculated by solving the LP in Equation 1. Below, taking the TSP as given by a weighted fully connected graph, we use the notation $\delta(S)$ for the set of edges joining locations in S to locations in $N \setminus S$.

$$\begin{array}{ll} \max \left(2 \sum_{S \subseteq N \ S \neq \emptyset} w_S\right) & \\ s.t. & \\ w_S \geq 0 & \forall S \subseteq N \ S \not\equiv \emptyset \\ \sum_{ij \in \delta(S)} w_S \leq d_{ij} & \forall i, j \in N \end{array} \quad (1)$$

The dual of this LP corresponds to the well-known Held-Karp relaxation of the TSP, which can be solved in polynomial-time.

Once a small set of non-zero w_S terms are computed as per Equation 1, a *nested* packing is obtained by following the post-processing procedure described by Özener, Ergun, and Savelsbergh 2013. A packing is *nested* if and only if $\forall S', S''$ s.t. $w_{S'} > 0$ and $w_{S''} > 0$, if $S' \cap S'' \not\equiv \emptyset$ then either $S' \subseteq S''$ or $S'' \subseteq S'$. For any optimal solution to Equation 1 there is a corresponding nested packing with the same objective value (Cornuéjols, Naddef, and Pulleyblank 1985). The nested constraint is required and, intuitively, it prevents overcharging a subset of locations that coalesce in a moat – i.e. prevents the allocation from violating the universally quantified constraint in the definition of the core. For the nesting critera to be violated there must be three distinct non-empty sets of locations S, S' and S'', so that $w_{S \cup S'} > 0$ and $w_{S' \cup S''} > 0$. Post-processing iteratively identifies and eliminates such cases. Identification is straightforward. For each elimination we take the assignment $\tau \leftarrow \min\{w_{S \cup S'}, w_{S' \cup S''}\}$, and make the following assignment updates to the moat widths: $w_S \leftarrow w_S + \tau$, $w_{S''} \leftarrow w_{S''} + \tau$, $w_{S \cup S'} \leftarrow w_{S \cup S'} - \tau$, and $w_{S' \cup S''} \leftarrow w_{S' \cup S''} - \tau$. This iterative procedure terminates yielding a nested packing, however the algorithm can take exponential time in the worst case. That being said, in all our experiments we found that nesting takes only a fraction of a second. Finally, an ε-core allocation is obtained where, for each $S \subseteq N$ we distribute the cost $3 \times w_S$ arbitrarily to the locations in the set $(N \setminus 0) \setminus S$ – distributing evenly to all nodes outside the moat for S, excluding the depot node 0.

Hybrid Proxy

Early on in our experimentation, we made an important observation that lead us to develop a sixth "blended" proxy, ϕ^{BLEND}. This proxy is a linear combination of ϕ^{MOAT} and ϕ^{DEPOT}. We experimentally identify a $\lambda \in [0,1]$ for which $\lambda \times \phi^{\text{MOAT}} + (1-\lambda) \times \phi^{\text{DEPOT}}$ provides an improved proxy for ϕ^{SV} compared to either component proxies in isolation. Experimentally we found $\lambda = 0.6$ to be most effective in Synthetic games. Our observation is that the ϕ^{MOAT} does not properly distribute the depot allocation of moat widths to other locations. In order to stay within the $1/2$-core allocation, that width is distributed in equal parts to all locations. Blending the ϕ^{MOAT} with ϕ^{DEPOT} mitigates this problem, and as we observe, increases proxy accuracy relative to ϕ^{SV}. The value of the improvement seems to decrease gradually as the size of games increases.

Analysis of Naïve Proxies

We refer to the three proxies ϕ^{DEPOT}, ϕ^{SHORT} and ϕ^{REROUTE}, as being *naïve*. Contrastingly, we call ϕ^{CHRIS}, ϕ^{MOAT} and ϕ^{BLEND} the *sophisticated* proxies. The formulation of the naïve proxies ϕ^{DEPOT} and ϕ^{SHORT} make them amenable to direct analysis of their worst case performance. We observe that ϕ^{DEPOT} and ϕ^{SHORT} can preform arbitrarily badly over or under estimating ϕ^{SV} in degenerate cases. The proofs are omitted for space.

Empirical Study

We implemented each of the six proxies discussed, along with a version of *ApproShapley* that uses *Concorde* (Applegate et al. 2007) to evaluate the characteristic function of the TSG. The *Concorde* program is used to find optimal solutions to TSPs. Rather than calculating ϕ^{SV} by direct enumeration as a baseline to compare proxies, we estimate that value using *ApproShapley* with *Concorde*. For the size of games we have considered, we find that 4000 iterations of *ApproShapley* gives accurate baseline values.

We experimented using a corpus of games comprised of two sets of TSGs. The first set of games are Synthetic. For each $i \in [4, \ldots, 35]$, we generate 20 instances of the Euclidean TSG with i locations occurring uniformly at random in a square of dimension $1,000$. The horizontal and vertical coordinates of the locations are represented using 32-bit floating point numbers. Those Euclidean games are available online at `http://users.cecs.anu.edu.au/~charlesg/tsg_euclidean_games.tar.gz`.

The second set of games is taken from large Real-World VRPs in the cities of Auckland, New Zealand; Canberra, Australia; and Sydney, Australia. Heuristic solutions to those VRPs are calculated using the *Indigo* solver (Kilby and Verden 2011). To give an indication of the scale and difficulty of these VRPs, the Auckland model comprises $1,166$ locations to be served using a fleet of at most 25 vehicles over a 7 day period. In the heuristic solution we collect tours of length 10 and 20 to created TSGs for testing. Because Real-World distance matrices are asymmetric, in all cases asymmetry is negligible, we induce symmetric problems by resolving for the greater of d_{ij} and d_{ji} – i.e. setting $d_{ij} = d_{ji} = \max\{d_{ij}, d_{ji}\}$. It total we obtain 69 Real-World games of size 10 and 44 games of size 20.

All experiments reported here were performed on a computer with an Intel i7-2720QM CPU running at 2.20GHz,

with 8GB of RAM, and running the *Ubuntu 12.04.3 LTS* operating system. For Synthetic problems with 35 locations, 4000 iterations of *ApproShapley* with exact TSP evaluations using *Concorde* (Applegate et al. 2007) takes 545 seconds. Computing ϕ^{CHRIS}, which replaces the exact TSP computation with an evaluation of the Christofides heuristic, results in a reduction to 11.39 seconds in total. Computing ϕ^{MOAT} takes under 1 second. All the naïve proxies (ϕ^{DEPOT}, ϕ^{SHORT}, and ϕ^{REROUTE}) take fractions of a second to compute.

To evaluate how well proxies perform in approximating ϕ^{SV} we measure the *point-wise root-mean-squared error* (RMSE) in each game. We also use Kendall's τ (1938) (written KT) to compare the ranking—i.e. least expensive to most expensive—of locations induced by the Shapley allocation and our proxies. The value τ measures the amount of disagreement between two rankings. It is customary to report τ as a normalized value (correlation coefficient) between 1 and -1, where $\tau = 1$ means that two lists are perfectly correlated (equal) and $\tau = -1$ means that two lists are perfectly anti-correlated (they are equal if one list is reversed).

Synthetic Data

Figure 2 shows the average root mean squared error and average KT distance for each proxy from ϕ^{SV} for all game sizes of the Synthetic data. We describe highlights of our results here. Overall, the best performing proxy is ϕ^{BLEND}, both in terms of lowest RMSE and highest average τ. The ϕ^{SHORT} and ϕ^{REROUTE} proxies are by far the worst, particularly in terms of approximating Shapley value, but also in terms of the ranking induced by the corresponding allocations. The computationally more expensive proxy ϕ^{REROUTE} always dominates ϕ^{SHORT}; a trend which continues throughout our testing on Real-World data as well. The proxy ϕ^{DEPOT} performs poorly at ranking, however does surprisingly well at approximation being almost competitive with the more sophisticated proxies. In ranking locations, ϕ^{REROUTE} regularly identifies the location ranked most costly according to the Shapley value, outperforming all proxies on this task for the synthetic data. More generally, in $\geq 60\%$ of synthetic games the ϕ^{CHRIS}, ϕ^{MOAT}, ϕ^{REROUTE}, and ϕ^{BLEND} proxies each identify the most costly location.

In the majority of the synthetic games, our analysis of rankings using Kendall's τ strongly implies that ϕ^{CHRIS}, ϕ^{MOAT} and ϕ^{BLEND} rankings are correlated with ϕ^{SV}. Put simply, we are confident that sophisticated proxies are inducing a ranking that is similar to the one induced by the Shapley value. They also reliably identify the most expensive location. Among the pure proxies, the ϕ^{CHRIS} proxy outperforms all the others at ranking by a slim margin. For example, it is able to identify the most expensive location according to the Shapley value 66.4% of the time. Additionally, regardless of the number of locations, the mean value for τ between ϕ^{SV} and ϕ^{CHRIS} is ≥ 0.55, and in *every* instance with 18 or more locations (and for the majority of instances between 4 and 17 locations) there is a statistically significant result for τ. Comparatively, ϕ^{BLEND} returns similar (and often higher) results for τ while achieving a statistically significant correlation with the ranking induced by ϕ^{SV} for every synthetic game instance with more than 8 players, save 6. The τ analysis in the case of ϕ^{MOAT} is less positive, gives strong correlation in instances with more than 20 locations, though still better than any of the naive proxies.

Our experimental analysis also considered how the types of allocation error differ between proxies. For example, we considered questions, such as: Do the proxies make a lot of small errors for low cost locations, or do they make large errors for locations that are apportioned large costs? Knowledge about the type and severity of errors made by our different proxies provides some guidance to the situations where we should have confidence in proxy allocations and/or the induced rankings.

Figure 3 shows the absolute error between all of the proxies and ϕ^{SV} graphed as a function of the allocation according to ϕ^{SV} for Synthetic games with 20 locations. For all the proxies, there appears to be a strong linear component to the error — many of the proxies allocate proportionally more (or less) cost compared to the ϕ^{SV} allocation. In some cases ϕ^{REROUTE} allocates more than 20-times the cost allocation by ϕ^{SV}, though typically this happens in the case of locations that received less than 10% of the Shapley allocation. We find that better performing proxies make more constant real-valued errors across all locations, regardless of actual allocation. The scatterplots for ϕ^{BLEND} and ϕ^{CHRIS} both show the weakest linear bias, with ϕ^{BLEND} showing a somewhat sub-linear bias. For example, ϕ^{CHRIS} and ϕ^{MOAT} can allocate 6-times ϕ^{SV}, though this only occurs in the case of locations whose Shapley allocation is less that 5% of the tour cost. Measuring the factor by which it overestimates allocations, the ϕ^{DEPOT} proxy appears to perform rather well, allocating at most 2.5-times the fair cost. The caveat is that ϕ^{DEPOT} is indiscriminate, also making proportionately large over-allocation errors to locations which are costly according to ϕ^{SV}.

Real-World Data

Measuring the performance of proxies in Real-World data from Auckland, Canberra, and Sydney, overall we find the quality of allocation is slightly degraded compared to measurements we made in synthetic games. We identified no significant performance differences between cities; we report aggregated statistics in this section. Table 1 summarizes results for 20 location games head-to-head between the Real-World and Synthetic datasets.

Examining the change in performance of sophisticated proxies when moving from the Synthetic to Real-World scenarios, the average RMSE increases from ≈ 0.075 to ≈ 0.153 while the average τ decreases from ≈ 0.63 to ≈ 0.28. Measuring RMSE, the degradation in performance of ϕ^{MOAT} is clearly the most sever. Measuring ranking error via τ, ϕ^{MOAT} degrades more gracefully compared to either ϕ^{CHRIS} or ϕ^{BLEND}. Measuring all proxy performances using RMSE, ϕ^{SHORT} is always dominated by ϕ^{REROUTE}, which in turn is strictly dominated by the sophisticated proxies. It is worth noting that in Real-World scenarios ϕ^{REROUTE} strictly dominates all the other proxies in its ability to identify the most costly location. In that regard ϕ^{MOAT} is a close second. Treating ranking error, Table 1 shows that ϕ^{REROUTE} actually performs comparably with best sophisticated proxy, ϕ^{MOAT},

	Synthetic		Real-World		Synthetic		Real-World			
	RMSE	St. Dev.	RMSE	St.Dev.	τ	St. Dev.	τ	St. Dev.	Synth.	Real.
Shortcut Distance	0.2965	0.0543	0.3239	0.1064	-0.0363	0.1358	0.0798	0.1891	5.0%	15.9%
Re-routed Margin	0.1826	0.0442	0.2902	0.0934	0.3813	0.1505	0.3476	0.1993	65.0%	65.9%
Depot Distance	0.0864	0.0182	0.0870	0.0303	0.5053	0.1464	0.1622	0.2313	45.0%	38.6%
Moat Packing	0.0758	0.0174	0.1969	0.0883	0.5304	0.1180	0.3064	0.1710	50.0%	61.4%
Christofides	0.0622	0.0136	0.0863	0.0311	0.5965	0.0999	0.3509	0.2258	45.0%	56.8%
60/40 Moat/Depot	0.0529	0.0084	0.0765	0.0301	0.6690	0.1105	0.2531	0.2487	50.0%	54.5%

Table 1: Comparison of performance between Synthetic and Real-World datasets for games with 20 locations. There are 20 games in the Synthetic corpus and 44 in the Real-World corpus. Root Mean Squared Error (RMSE) and Standard Deviation (St. Dev.) are reported on the left where lower is better. In the center average KT distance (τ) and Standard Deviation (St. Dev.) is reported where higher is better; $+1$ means the two lists are perfectly correlated and -1 means the two lists are anti-correlated. The percentage of most expensive locations according to the Shapley ordering correctly identified by the proxies is on the right.

in terms of τ. Repeating our observations for the synthetic corpus, in the Real-World games the sophisticated proxies have a greater percentage of statistically significant results for τ. For a majority of the instances, ϕ^{CHRIS} and ϕ^{MOAT} achieve a statistically significant correlation with the ranking induced by ϕ^{SV}. Table 1 shows a side by side comparison of the games with 20 locations for the Real-World and Synthetic data. Moving from synthetic to Real-World we see the performance of ϕ^{CHRIS} and ϕ^{BLEND} noticably degrade, though they do continue to achieving fairly low RMSE scores. Again, note that all sophisticated proxies are also good at identifying the most costly location.

Examining Real-World games with 20 locations, Figure 3 gives the error scatter plots for all proxies as a function of allocation according to ϕ^{SV}. There is an observable linear component to the error for several of the proxies for the Synthetic dataset. There is a more uniform distribution of errors among locations in the Real-World data. This is evidenced by the pillar like shapes for most of the plots; demonstrating that in the Real-World data, many of the ϕ^{SV} allocations cluster around a uniform allocation of around 5–8%. Indeed, the observed tight clustering of actual Shapley values explains the respectable performance of ϕ^{DEPOT} in the Real-World datasets. The much taller shapes we see in Figure 3 for the Real-World data indicate that proxy errors are more randomly distributed among Real-World locations, and that in Real-World scenarios proxies make proportionately larger allocation errors irrespective of the actual ϕ^{SV} allocation.

Conclusions and Future Work

We studied the problem of fairly apportioning costs in transportation scenarios, specifically TSGs. The Shapley value is a highly appealing division concept for this task. Since it cannot be evaluated in reasonable time, we considered a number of proxies for the Shapley value. We examined proxy performance both in terms of approximating the Shapley value and the ranking of locations induced by the Shapley value. The stand-out proxies with respect to both measures are ϕ^{CHRIS} and ϕ^{BLEND}, a mixture of ϕ^{DEPOT} and ϕ^{MOAT}. These proxies can be computed in reasonable time, and exhibit good properties in both synthetic Euclidean games and real-world transportation scenarios.

Extensions of our work should develop proxies for the more general setting of vehicle routing games, to quantify the importance of agent synergies that are unique to the multi-vehicle model. The transport companies we interact with further seek to understand the impact of time windows (both the duration and position of allowable service times), and the effect of delivery frequency on allocated costs. Thus, a highly motivated and rich variety of problems is available to be considered for future work.

Acknowledgments

NICTA is funded by the Australian Government through the Department of Communications and the Australian Research Council through the ICT Centre of Excellence Program. Casey Cahan was supported by an Summer Research Scholarship at The Australian National University. Toby Walsh also receives support from the Asian Office of Aerospace Research and Development (AOARD 124056) and the German Federal Ministry for Education and Research through the Alexander von Humboldt Foundation.

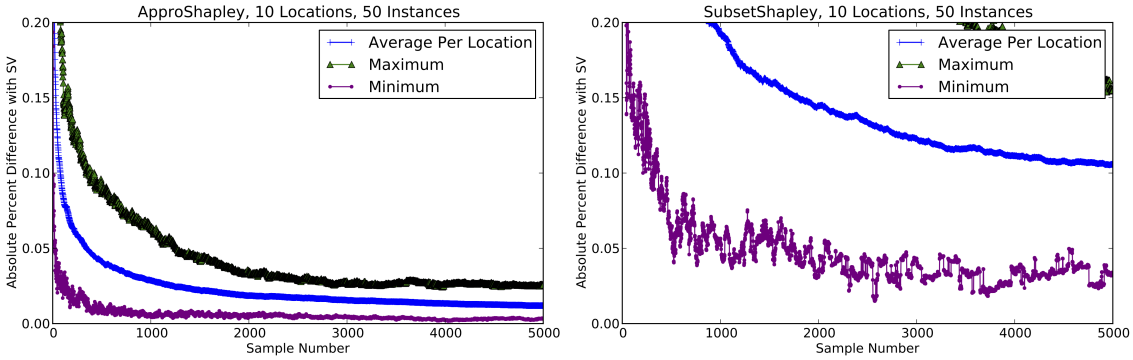


Figure 1: Comparison of the performance of ApproShapley (left) and SubsetShapley(right) over 5000 iterations for TSGs with 10 locations. The graphs show minimum and maximum (outside the graph range for SubsetShapley) error in the Shapley value for a single location averaged over 50 instances. The error is computed as a percentage difference between the actual Shapley value and the one computed by sampling. Additionally, the average percent error for all locations per iteration is shown.

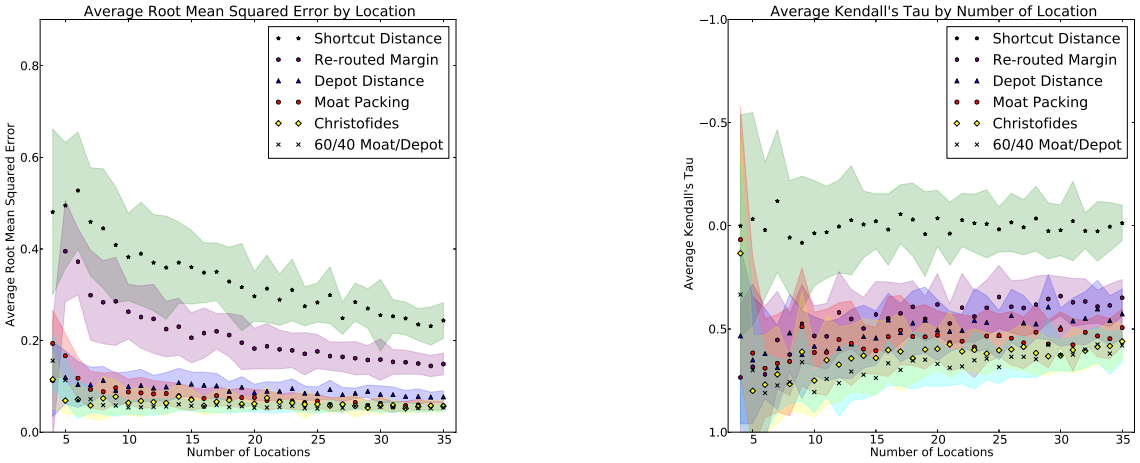
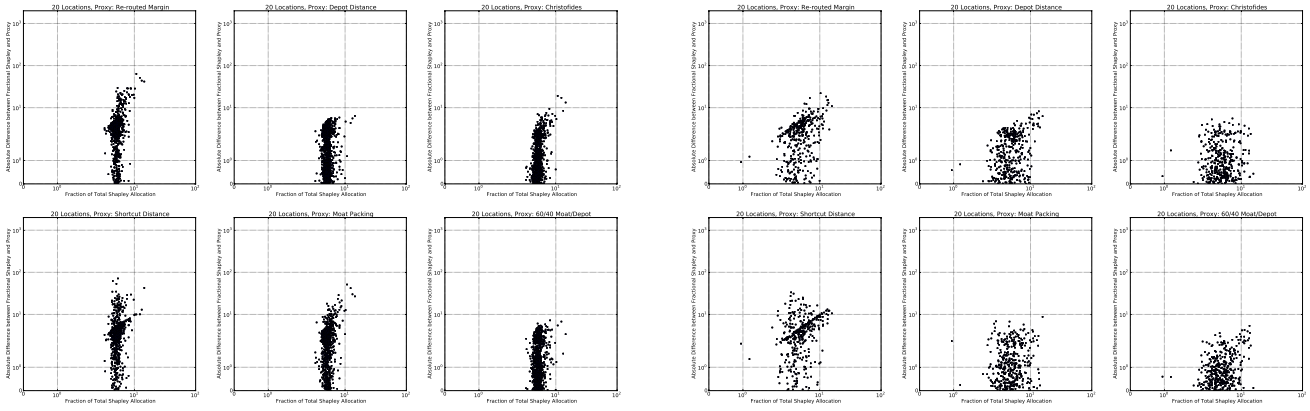


Figure 2: Performance of the five pure proxies and one hybrid proxy according to: (**left**) RMSE averaged over the 20 games generated for each number of locations, and (**right**) Kendall's tau rank correlation averaged over the 20 games generated for each number of locations. The error bands correspond to plus or minus one standard deviation. The vertical axis of our Kendall's tau plot has been inverted for ease of comparison – more correlated lists are towards the bottom of the graph.



Real-World Data. Synthetic Data.

Figure 3: Absolute value of the difference between the ϕ^{SV} and ϕ^{PROXY} plotted as a function of ϕ^{SV} for all games in the Synthetic and Real-World datasets with 20 locations.

References

Applegate, D. L.; Bixby, R. E.; Chvatal, V.; and Cook, W. J. 2007. *The traveling salesman problem: a computational study*. Princeton University Press.

Bachrach, Y.; Markakis, E.; Resnick, E.; Procaccia, A. D.; Rosenschein, J. S.; and Saberi, A. 2010. Approximating power indices: theoretical and empirical analysis. *Autonomous Agents and Multi-Agent Systems* 20(2):105–122.

Castro, J.; Gómez, D.; and Tejada, J. 2009. Polynomial calculation of the Shapley value based on sampling. *Comput. Oper. Res.* 36(5):1726–1730.

Chalkiadakis, G.; Elkind, E.; and Wooldridge, M. 2011. Computational aspects of cooperative game theory. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 5(6):1–168.

Christofides, N. 1976. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, DTIC Document.

Cornuéjols, G.; Naddef, D.; and Pulleyblank, W. 1985. The traveling salesman problem in graphs with 3-edge cutsets. *Journal of the ACM (JACM)* 32(2):383–410.

Curiel, I. 2008. Cooperative combinatorial games. In Chinchuluun, A.; Pardalos, P.; Migdalas, A.; and Pitsoulis, L., eds., *Pareto Optimality, Game Theory And Equilibria*, volume 17 of *Springer Optimization and Its Applications*. Springer New York. 131–157.

Faigle, U., and Kern, W. 1993. On some approximately balanced combinatorial cooperative games. *ZOR Methods and Models of Operations Research* 38(2):141–152.

Faigle, U.; Fekete, S.; Hochstättler, W.; and Kern, W. 1998. On approximately fair cost allocation in euclidean tsp games. *Operations-Research-Spektrum* 20(1):29–37.

Fatima, S. S.; Wooldridge, M.; and Jennings, N. R. 2007. A randomized method for the Shapley value for the voting game. In *Proceedings of the 6th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 07)*, 157–165.

Fatima, S. S.; Wooldridge, M.; and Jennings, N. R. 2008. A linear approximation method for the Shapley value. *Artificial Intelligence* 172(14):1673–1699.

Garey, M. R., and Johnson, D. S. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York: W.H. Freeman.

Golden, B. L.; Raghavan, S.; and Wasil, E. A. 2008. *The Vehicle Routing Problem: Latest Advances and New Challenges: latest advances and new challenges*, volume 43. Springer.

Held, M., and Karp, R. M. 1962. A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial & Applied Mathematics* 10(1):196–210.

Kendall, M. G. 1938. A new measure of rank correlation. *Biometrika* 30(1/2):81–93.

Kilby, P., and Verden, A. 2011. Flexible routing combing constraint programming, large neighbourhood search, and feature-based insertion. In *2nd Workshop on Artificial Intelligence and Logistics*. Barcelona, Spain.

Leech, D. 2003. Computing power indices for large voting games. *Management Science* 49(6):831–837.

Maleki, S.; Tran-Thanh, L.; Hines, G.; Rahwan, T.; and Rogers, A. 2013. Bounding the estimation error of sampling-based Shapley value approximation with/without stratifying. *CoRR* abs/1306.4265.

Mann, I., and Shapley, L. S. 1960. Values for large games IV: Evaluating the electoral college by monte carlo. Technical report, The RAND Corporation, Santa Monica, CA, USA.

Owen, G. 1972. Multilinear extensions of games. *Management Science* 18(5/2):64–79.

Özener, O. O.; Ergun, O.; and Savelsbergh, M. 2013. Allocating cost of service to customers in inventory routing. *Oper. Res.* 61(1):112–125.

Papadimitriou, C. 1994. *Computational Complexity*. Addison-Wesley Publishing Company, Inc.

Peleg, B., and Sudhölter, P. 2007. *Introduction to the Theory of Cooperative Games*. Springer.

Potters, J. A.; Curiel, I. J.; and Tijs, S. H. 1992. Traveling salesman games. *Mathematical Programming* 53(1-3):199–211.

Shapley, L. S. 1953. A value for n-person games. In Kuhn, H., and Tucker, W. W., eds., *Contributions to the Theory of Games*, volume 2 of *Annals of Mathematical Studies*. Princeton University Press.

Tamir, A. 1989. On the core of a traveling salesman cost allocation game. *Operations Research Letters* 8(1):31 – 34.

Winter, E. 2002. The Shapley value. In *Handbook of Game Theory with Economic Applications*. Elsevier. chapter 53, 2025–2054.

Young, H. P. 1985. Monotonic solutions of cooperative games. *International Journal of Game Theory* 14(2):65–72.