GROSS SUBSTITUTABILITY : AN ALGORITHMIC SURVEY

RENATO PAES LEME*

Abstract. The concept of gross substitute valuations was introduced by Kelso and Crawford as a sufficient conditions for the existence of Walrasian equilibria in economies with indivisible goods. The proof is algorithmic in nature: gross substitutes is exactly the condition that enables a natural price adjustement procedure – known as *Walrasian tatônnement* – to converge to equilibrium.

The same concept was also introduced independently in other communities with different names: $M^\natural$-concave functions (Murota and Shioura), Matroidal and Well-Layered maps (Dress and Terhalle) and valuated matroids (Dress and Wenzel). Here we survey various definitions of gross subtitutability and show their equivalence. We focus on algorithmic aspects of the various definitions. In particular, we highlight that gross substitutes are the exact class of valuations for which demand oracles can be computed via an ascending greedy algorithm. It also corresponds to a natural discrete analogue of concave functions: local maximizers correspond to global maximizers.

Finally, we discuss algorithms for the welfare problem (computing an optimal allocation of a set of items when agents have gross substitute valuations) as well as the related problem of computing Walrasian prices. We discuss approximation schemes based on the tatônnement procedure, linear programming approaches and purely combinatorial strongly-polynomial time algorithms.

1. Gross substitutes and Walrasian tatônnement.

The notion of *gross substitutes* was introduced by Kelso and Crawford [16] in order to analyze two sided matching markets of workers and firms. Originally it was defined as a condition on the *gross product* generated by a set of workers for a given firm, hence the name *gross substitutes*. Such condition allowed a natural salary adjustment process to converge to a point where each worker is hired by some firm and no worker is over-demanded. Gul and Stacchetti [11] later use the same notion to analyze the existence of price equilibria in markets with indivisible goods. For this survey, we adopt the Gul and Stacchetti terminology and talk about buyers/items/prices instead of firms/workers/salaries as in Kelso and Crawford.

Before we proceed, we fix some notation: we denote by $[n] = \{1, \ldots, n\}$ a set of items (goods). A *valuation* over such items is a function $v : 2^{[n]} \to \mathbb{R}$ such that $v(\emptyset) = 0$[1]. Given a price vector $p \in \mathbb{R}^n$ and a set $S \subseteq [n]$, we denote $p(S) = \sum_{j \in S} p_j$. We will define $v_p(S) = v(S) - p(S)$ as the value of a subset S under the price vector p. This corresponds to the utility of an agent with this valuation for acquiring such set under those prices. Given disjoint sets S, T we define the *marginal value* of T with respect to S as $v(T|S) = v(T \cup S) - v(S)$. We sometimes omit braces in the representation of sets when this is clear from the context, for example, by $v(i, j|S)$ we denote $v(\{i, j\}|S)$.

An *economy with indivisible goods* is composed by a set $[n]$ of items (goods) and $[m]$ of buyers (agents) where each agent $i \in [m]$ has a valuation $v^i : 2^{[n]} \to \mathbb{R}$. We use the notion of the *demand correspondence* to define an equilibrium of this economy:

DEFINITION 1.1 (demand correspondence). *Given a valuation function $v : 2^{[n]} \to \mathbb{R}$ and a vector of prices $p \in \mathbb{R}^n$, we define the demand correspondence as the family of sets that maximize the utility of an agent under a price vector p :*

$$D(v, p) := \{S \subseteq [n]; v_p(S) \geq v_p(T), \forall T \subseteq [n]\}$$

*Google Research NYC, (renatoppl@google.com). Part of this work was done while the author was a post-doc at Microsoft Research Silicon Valley.

[1]Note that we don't require monotonicity in the definition. When we refer to a valuation for which $v(S) \leq v(T)$ whenever $S \subseteq T$, we will refer to it as a *monotone* valuations or valuations statifying *free-disposal*.

DEFINITION 1.2 (Walrasian equilibrium). *Given an economy with indivisible goods with n goods, m agents and valuations $\{v^i\}_i$ satisfying free-disposal*[2]*, a* Walrasian equilibrium *corresponds to a vector of prices $p \in \mathbb{R}^n_+$ and a partition of the goods in disjoint sets $[n] = \cup_{i=1}^m S_i$ such that $S_i \in D(v^i, p)$ for all i.*

A reader familiar with the duality theorem in linear programming will readily recognize that the definition of Walrasian equilibrium closely resembles the *complementarity conditions* where the prices play the role of dual variables. Indeed, this is formalized by the results known as the First and Second Welfare Theorem. The First Welfare Theorem states that if $(p, S_1, \ldots, S_m)$ is a Walrasian equilibrium then this partition corresponds to the optimal allocation of goods, i.e., the allocation maximizing $\sum_i v^i(S_i)$. The proof is quite elementary: let $S_1^*, \ldots, S_m^*$ be any partition maximizing the welfare. Then since $S_i \in D(v^i, p)$, it must be the case that: $v^i(S_i) - p(S_i) \geq v^i(S_i^*) - p(S_i^*)$. Summing for all i and observing that $\sum_i p(S_i) = p([n]) = \sum_i p(S_i^*)$, we conclude that $\sum_i v^i(S_i) \geq \sum_i v^i(S_i^*)$.

The analogy with linear programming is completed by what is called the Second Welfare Theorem. It states that if $(p, S_1, \ldots, S_m)$ is a Walrasian equilibrium and $S_1^*, \ldots, S_m^*$ maximizes $\sum_i v^i(S_i^*)$, then $(p, S_1^*, \ldots, S_m^*)$ is also a Walrasian equilibrium. The proof is also simple, observe that summing $v^i(S_i) - p(S_i) \geq v^i(S_i^*) - p(S_i^*)$ for all i we obtain $\sum_i v^i(S_i) \geq \sum_i v^i(S_i^*)$. But since this is an equality, we should have an equality for each agent i: $v^i(S_i) - p(S_i) = v^i(S_i^*) - p(S_i^*)$, hence $S_i^* \in D(v^i, p)$.

A natural question is for which economies there exist Walrasian equilibria. Kelso and Crawford define a very natural price adjustment procedure and define gross substitutes as the natural sufficient condition for such process to converge. The general idea behind this procedure goes back to Walras' *tatônnement procedure* [32], where tatônnement means *trial-and-error*. The idea is that we start with an arbitrary price vector and compute one set in the demand of each agent. Then, for each item that is demanded by more then one agent (overdemanded) we increase the price. For each item that is demanded by no agent (underdemanded), we decrease the price. We iterate this until no item is overdemanded or underdemanded.

Let's describe this procedure precisely. We will make some modifications to the idea above to make the procedure simpler to analyze. Instead of starting from an arbitrary price vector p, we will start with zero prices for all items and only allow prices to increase. Moreover, we will start with all the items allocated to the first player at zero price and we will take turns asking buyers to choose their favorite set of items given prices as follows: the current price p_j for items currently allocated to him and $p_j + \delta$ for items allocated to other players. Once he takes items from other players, the prices of such items increase by δ.

Notice that the procedure has to stop at some point, since prices cannot increase indefinitely. If the price of an item is higher then $\max_{i,S} v^i(S)$, for example, no agent will demand this item and the price will freeze. Let p be the final price and p^i be the price faces by each agent. It should be the case that $S_i \in D(v^i, p^i)$, which means that for all $T \subseteq [n]$, $v^i(S_i) - p^i(S_i) \geq v^i(T) - p^i(T)$. This can be re-written as: $v^i(S_i) - p(S_i) \geq v^i(T) - p(T) - \delta|T \setminus S_i|$.

[2]The definition of Walrasian equilibrium can be changed to incorporate valuations not satisfying free-disposal. We do so by partititioning the items in $m+1$ disjoint sets $S_0, S_1, \ldots, S_m$. The items in S_0 are not allocated and are required to be priced at zero at in equilibrium.

ALGORITHM 1: Walrasian tatônnement procedure

Input: $\delta > 0$, $n, m \in \mathbb{Z}_+$ and v^i for $i \in [m]$
Set zero prices for all items: $p_j = 0, \forall j \in [n]$
Set initial allocation $S_1 = [n], S_i = \emptyset, \forall i \in [m] \setminus \{1\}$
Implicitely define $p^i \in \mathbb{R}^n$ as a function of p s.t. $p^i_j = p_j$ if $j \in S_i$ and $p^i_j = p_j + \delta$ o.w.
while there exists i such that $S_i \notin D(v^i, p^i)$
    find a demanded set under the p^i price vector $X_i \in D(v^i, p^i)$
    update prices: for $j \in X_i \setminus S_i$, set $p_j = p_j + \delta$ (vectors p^i are implicitly updated)
    update allocations: $S_i = X_i$ and $S_j = S_j \setminus X_i$ for $j \neq i$

In the limit as $\delta \to 0$, we recover a price vector and allocation such that $v^i(S_i) - p(S_i) \geq v^i(T) - p(T)$. To make the previous statement precise, let $(p^t, S^t_1, \ldots, S^t_m)$ be the outcome of the Walrasian tatonnment procedure for $\delta_t = \frac{1}{t}$ for $t \in \mathbb{Z}_+$. Since there are finitely many allocations $(S^t_1, \ldots, S^t_m)$, there is one allocation that happens infinitely often. Let $S_1, \ldots, S_m$ be such allocation and let $t_1 < t_2 < \ldots$ be the infinite subsequence corresponding to this allocation. Since p^t is bounded, passing to a subsequence if necessary, we can assume that $p^t \to p$. So taking $t \to \infty$ for this subsequence, we get $v^i(S_i) - p(S_i) \geq v^i(T) - p(T)$ for all i and $T \subseteq [n]$.

The argument above gives us an existential proof of a price vector p and an allocation S_i such that each agent is getting his optimal bundle under the current prices. This is not yet a Walrasian equilibrium, since Definition 1.2 requires the allocation to be a partition of the set of items, i.e., $\cup_i S_i = [n]$. The definition of gross substitutability is exactly what is needed to ensure that we can run the procedure above in such a way that all the items are allocated in the end. Since we started the Walrasian tatônnement procedure with a partition of the items, if we can always find a demanded set $X_i \in D(v^i, p^i)$ containing his currently allocated items, i.e., $S_i \subseteq X_i$, then we can guarantee the invariant that no item is even un-allocated during the execution of the algorithm. This motivates the following definition:

Definition 1.3 (gross substitutes, Kelso and Crawford [16]). *A valuation function satisfies the gross substitutes property if for any price vectors $p \in \mathbb{R}^n$ and $S \in D(v, p)$, if p' is a price vector with $p \leq p'$, then there is a set $S' \in D(v, p')$ such that $S \cap \{j; p_j = p'_j\} \subseteq S'$.*

In other words: if an agent with a gross substitute valuation demands a set S of items under a price vector p and the price of some items subsequently increase, the agent still has a demanded set that contains the items in S whose price didn't increase.

Theorem 1.4 (Kelso and Crawford [16]). *If valuations $v^1, \ldots, v^m$ sastisfy the gross substitutes property, then a Walrasian equilibrium always exists.*

In some sense, gross substitutability is also necessary for the existence of Walrasian equilibria. Gul and Stacchetti [11] show the following: let $\mathcal{C}$ be a class of valuation functions that contains all *unit demand valuations*, i.e., all valuations of the type $v(S) = \max_{j \in S} v(\{j\})$. Then if $\mathcal{C}$ is such that for all $v^1, \ldots, v^m \in \mathcal{C}$ there is a Walrasian equilibrium, then all valuations in $\mathcal{C}$ are gross substitutes.

2. Examples, Non-Examples and Submodularity. It is instructive to have in mind a couple of examples and non-examples of gross substitute functions, to guide

our intuition in the following sections. Three simple classes of valuations that can be readily recognized as gross substitutes from Definition 1.3 are:

1. *additive valuations*, i.e., valuations for which $v(S) = \sum_{i \in S} v(\{i\})$
2. *unit-demand valuations*, i.e., valuations for which $v(S) = \max_{i \in S} v(\{i\})$
3. *symmetric concave valuations*, i.e., valuations of the form $v(S) = f(|S|)$ for some monotone concave function $f : \mathbb{R}_+ \to \mathbb{R}_+$.

A less obvious example is the class of valuations known as *assignment valuations* introduced by Shapley [31], also called OXS valuations by Lehmann, Lehmann and Nisan [21]. An assignment valuation with n items can be represented as an $n \times k$ matrix V with non-negative entries, where each row corresponds to an item and each column to a position. Entry $V_{i,j}$ corresponds to the value of item i in position j. Each item can be assigned to a single position and each position can be assigned only one item. The value of a subset S of items is the value of the optimal assignment of items in S to positions. Items and positions are allowed to be left unassigned. Mathematically:

$$v(S) = \max\{\textstyle\sum_{i,j} V_{ij} \cdot x_{ij} \text{ s.t. } \sum_k x_{kj} \leq 1, \sum_k x_{ik} \leq 1 \text{ and } x_{ij} \in \{0,1\}, \forall i, j\}$$

We postpone until Section 8 a proof that all assignment valuations are gross substitutes, but we remark that unit-demand functions are the special case where there is only one position, additive functions correspond to the case where the matrix is an $n \times n$ diagonal matrix and the symmetric concave valuations to the case in which all the rows of the matrix are equal.

Hatfield and Milgrom [13] define a generalization of assignment valuations, which they call *endowed assignment valuations*. The show that this class also satisfied the gross substitutes property and that it captures most practical applications of gross substitutes valuations in matching markets. A valuation v on n items is said to be an endowed assignment valuation if there is an assignment valuation w on a larger set of items such that $v(S) = w(S|T)$.

Another important example of gross substitutes is the class of *matroid rank functions*. This survey won't formally require any knowledge of matroid theory, basic familiarity with matroids will be useful to guide the reader's intuition later on. The topic is too extesive to survey here, but we point to Lawler [20], Oxley [28] or Schrijver [30] for a comprehensive discussion.

We remark that even though matroid rank functions are gross substitute valuations, sums of matroid rank functions might *not* be. For example, given three items $\{a, b, c\}$ define for each $i \in \{a, b, c\}$, the function $r^i : 2^{\{a,b,c\}} \to \mathbb{R}$ such that: $r^i(\emptyset) = 0$, $r^i(S) = 1$ for $|S| = 1$, $r^i(\{a, b, c\} \setminus i) = 1$, and $r^i(S) = 2$ for all remaining subsets S. Notice that for each i, r^i is a matroid rank function and hence satisfy the gross substitutability. However, the valuation $v = r^a + 2 \cdot r^b + 3 \cdot r^c$ does not satisfy gross substitutability. In order to see that, observe that for the price vector $p = [4, 5, 4]$, $D(v; p) = \{\{a\}, \{c\}, \{a, c\}, \{b, c\}\}$. If the price of item c increases to ∞, i.e., $p' = [4, 5, \infty]$, then demand set becomes $D(v; p') = \{\{a\}\}$. So the increase in price of item c makes item b no longer belong to any demanded set, violating Definiton 1.3.

Gul and Stachetti [11] observe that gross substitutes are a subclass of *submodular functions*:

DEFINITION 2.1 (submodularity). *A valuation function is said to be* submodular *if for all subsets $S, T \subseteq [n]$, $v(S \cap T) + v(S \cup T) \leq v(S) + v(T)$. Equivalently, for every $S \subseteq [n]$ and $i, j \notin S$, $v(i, j|S) \leq v(i|S) + v(j|S)$.*

THEOREM 2.2 (Gul and Stacchetti [11]). *Every gross substitute valuation function is submodular.*

Proof. Let v be a gross substitute valuation functions. Given $S \subseteq [n]$ and $i, j \notin S$, consider the price vector[3] such that $p_t = \infty$ for $t \notin S \cup \{i, j\}$, $p_t = -\infty$ for $t \in S \cup \{j\}$ and $p_i = v(i|S \cup \{j\})$. Clearly $S \cup \{i, j\} \in D(v, p)$. Now, if one defines p' such that $p'_j = \infty$ and $p'_t = p_t$ for all other t, then by gross substitutability, $S \cup \{i\}$ must be a demanded set. Therefore: $v(i|S) \geq v(i|S \cup \{j\})$. □

Since the example $v = r^a + 2 \cdot r^b + 3 \cdot r^c$ earlier in this section is the sum of matroid rank functions, and hence submodular, it is clear that gross substitutes is a strict subclass of submodular functions. Another good source of examples of submodular but not gross substitute functions comes from the class of budget additive functions. We say that a valuation function is budget additive if it is of the form: $v(S) = \min\{B, \sum_{i \in S} w_i\}$ for non-negative real numbers $B, w_1, \ldots, w_n$. The following example due to Lehmann, Lehmann and Nisan [21] shows function that is budget additive but not gross substitutes: consider three items $\{a, b, c\}$ with weights $w_a = w_b = 1$ and $w_c = 2$ and budget $B = 2$. In order to see that the associated budget additive function is not gross substitutes, notice that for the prices $p = [\frac{1}{2}, \frac{1}{2}, 1]$, $D(v, p) = \{\{a, b\}, \{c\}\}$, but if the price of a increases and the price vector becomes $p' = [1, \frac{1}{2}, 1]$, then the demand correspondence becomes $D(v, p') = \{\{c\}\}$, i.e., the increase in the price of a makes item b be no longer demanded.

3. Gross substitutes, greedy demand oracles and local search. An important property of gross substitute valuations is that it is the exact class of functions for which demand oracles can be computed via a greedy or local search algorithms. In this section we make this statement precise by defining *matroidal maps* and *discrete concave valuations*. The first part of the survey will be devoted to show that those three concepts are equivalent.

An algorithmic primitive needed to implement the Walrasian tatônnement procedure is the computation of a set in the demand correspondence $X \in D(v, p)$. This is usually referred as the *demand oracle problem*. A simple heuristic to compute demand oracles is the *greedy algorithm*: start with the empty set X and keep adding the element $j \notin X$ that gives the maximum improvement to $v_p(X)$. In other words:

DEFINITION 3.1 (matroidal map). *A valuation function is a* matroidal map *if for all price vectors $p \in \mathbb{R}^n$, the greedy algorithm implements a demand oracle, i.e., $\mathrm{G}(v, p) \in D(v, p)$, where $\mathrm{G}(v, p)$ is the output of Algorithm 2.*

THEOREM 3.2. *A valuation function satisfies the gross substitute property if and only if it is a matroidal map.*

Before we proceed with the task of proving the previous theorem, we also mention

[3] allowing prices to take values ∞ and $-\infty$ simplifies the arguments. To be more precise, one can view such prices as M or $-M$ for $M = 1 + \max_S v(S)$.

ALGORITHM 2: Greedy demand oracle

Input: $p \in \mathbb{R}^n_+$, $v : 2^{[n]} \to \mathbb{R}_+$
Initialize $X = \emptyset$
repeat
find $j^* \in [n] \setminus X$ maximizing $\Delta_j = v(j|X) - p_j$
if $\Delta_{j^*} > 0$, $X = X \cup \{j^*\}$
if $X = [n]$ or $\Delta_{j^*} \leq 0$, **return** X

another algorithmic definition of gross substitutability based on local search. Consider the following heuristic to compute demand oracles: start at an arbitrary set X and try to find a set improving v_p in the neighborhood $\mathcal{N}$ of X, where the neighborhood is composed by all sets that can be obtained from X by adding one element, removing one element or exchanging an element in the set by one element outside the set.

ALGORITHM 3: Local search demand oracle

Input: $p \in \mathbb{R}^n_+$, $v : 2^{[n]} \to \mathbb{R}_+$, $X_0 \subseteq [n]$
Initialize $X = X_0$
repeat
Let $\mathcal{N} = \{X \cup \{i\}; i \notin X\} \cup \{X \setminus \{i\}; i \in X\} \cup \{X \cup \{i\} \setminus \{i'\}; i \notin X, i' \in X\}$
If $\max_{Y \in \mathcal{N}} v_p(Y) \leq v_p(X)$, **return** X
Else choose some $Y \in \mathcal{N}$ with $v_p(Y) > v_p(X)$ and let $X = Y$.

Gul and Stacchetti [11] show that yet another way of defining gross substitutes is as the class of valuation functions for which local search is exact, i.e., it doesn't get stuck on local minima:

DEFINITION 3.3 (discrete concave valuation). *A valuation function is discrete concave if for all price vectors $p \in \mathbb{R}^n$, the local search algorithm implements a demand oracle, i.e.,* $\mathrm{L}(v, p, X_0) \in D(v, p)$, *where* $\mathrm{L}(v, p, X_0)$ *is the output of Algorithm 3.*

Equivalently, a valuation function is discrete concave if and only if for all price vectors $p \in \mathbb{R}^n$, $S \in D(v, p)$ iff $v_p(S) \geq v_p(S \cup i)$, $v_p(S) \geq v_p(S \setminus j)$ and $v_p(S) \geq v_p(S \cup i \setminus j)$, for all $i \notin S$ and $j \in S$.

THEOREM 3.4 (Gul and Stacchetti [11]). *A valuation function satisfies the gross substitute property if and only if it is discrete concave.*

4. A price independent local characterization. We make a brief detour and look at a different, yet very related question about gross substitutes. In the previous section we gave two alternative algorithmic characterizations of gross substitutes. All characterizations given so far involve prices, i.e., they are of the form: a valuation v satisfied the gross substitutes property if for all price vectors p, the pair (v, p) has some given property. The question of giving an explicit characterization of gross substitutes was resolved simultaneously by Fujishige and Yang [9] and Reijnierse, Gellekom and Potters [29]. The first paper provides a powerful connection to the theory of Discrete Convex Analysis, which we discuss in more detail in Section 7. We focus first on the definition given by Reijnierse et al [29].

THEOREM 4.1 (Reijnierse, Gellekom and Potters [29]). *A valuation function has the gross substitutes property iff it is submodular and for all sets $S \subseteq [n]$ and all distinct $i, j, k \notin S$, the following holds:*

$$v(i,j|S) + v(k|S) \leq \max\left[\ v(i|S) + v(j,k|S), v(j|S) + v(i,k|S)\ \right] \qquad \text{(RGP)}$$

Proof sketch. The high level picture of their proof is quite simple and illuminating. Here we provide a brief sketch of it. First, they show that a function doesn't have the gross substitutes property iff it is possible to find the following certificate: a price vector $p \in \mathbb{R}^n$ such that

$$\text{either (i) } D(v,p) = \{S, S \cup \{i,j\}\} \text{ or (ii) } D(v,p) = \{S \cup \{k\}, S \cup \{i,j\}\}.$$

Notice that the existence of such certificate clearly shows the violation of gross substitutability: the increase in the price of j would make the demand set become $\{S\}$ in case (i) and $\{S \cup \{k\}\}$ in case (ii). Therefore, item i would no longer be in the demand set. Proving the other direction requires more work, but its essence is to search for a minimal violation of gross substitutability by starting with an arbitrary one and changing the price vector so to shrink the demand set until it is minimal.

The second step is to transform the existence of certificates as above in simple conditions on v. This is based on two observations: There exists a certificate of type (i) iff the v is not submodular. There exists a certificate of type (ii) iff there is a violation of condition (RGP).

First, assume that we have a certificate of type (i). So, there are prices such that $0 = v(i,j|S) - p_i - p_j > \max[v(i|S) - p_i, v(j|S) - p_j]$. Summing the inequalities $0 > v(i|S) - p_i$ and $v(i,j|S) - p_i - p_j > v(j|S) - p_j$ we get: $v(i,j|S) > v(i|S) - v(j|S)$ which is a violation of submodularity. Conversely, if you have a violation of submodularity for i, j, S, take $p_t = -\infty$ for $t \in S$, $p_t = \infty$ for $t \notin S \cup \{i,j\}$ and $p_i = v(i|S) + \epsilon$ and $p_j = v(j|S) + \epsilon$ for some tiny ϵ and this gives us a certificate of type (i).

Assume now that we have a certificate of type (ii). So, there are prices such that $v(k|S) - p_k = v(i,j|S) - p_i - p_j > \max[v(i|S) - p_i, v(j|S) - p_j, v(i,k|S) - p_i - p_k, v(j,k|S) - p_j - p_k]$. Summing the inequalities such that the prices cancel, we get: $v(i,j|S) + v(k|S) > \max[v(i|S) + v(j,k|S), v(j|S) + v(i,k|S)]$. Conversely, if you have a violation of the condition (RGP) for i, j, k, S, let $\phi > 0$ be the value of the violation, i.e., $\phi = v(i,j|S) + v(k|S) - \max\{v(i|S) + v(j,k|S), v(j|S) + v(i,k|S)\}$. Now, consider prices $p_t = -\infty$ for $t \in S$, $p_t = \infty$ for $t \notin S \cup \{i,j,k\}$ and $p_i = v(i|S \cup \{j\}) - \frac{1}{2}\phi$, $p_j = v(j|S \cup \{i\}) - \frac{1}{2}\phi$ and $p_k = v(k|S) + v(i,j|S) - v(i|S) - v(j|S) - \phi$. It is straightforward to check that such prices give us a certificate of type (ii). □

Now we explain what we mean by a *local* characterization. Given a valuation function v and two sets $S, R \subseteq [n]$ we can define a restriction $v_{R|S} : 2^R \to \mathbb{R}$ by $v_{R|S}(T) = v(T|S)$. We say that this is a k-restriction if $|R| = k$. Observe that usual properties as monotonicity and submodularity are properties of the restrictions, i.e., a function is monotone iff each 1-restriction is monotone. A function is submodular iff each 2-restriction is submodular. A corollary of Theorem 4.1 is that:

COROLLARY 4.2. *A valuation function satisfies the gross substitutes property iff every 3-restriction satisfied the gross substitutes property.*

An equivalent characterization of the one in Theorem 4.1 was also observed in Lehmann, Lehmann and Nisan [21] and Bing, Lehmann and Milgrom [4]. The latter chracterization defines for each valuation v and subset $S \subseteq [n]$ a measure of how two goods i and j are substitutes to each other. Given $i, j \notin S$, let:

$$\alpha_S(i,j) = v(i|S) + v(j|S) - v(i,j|S)$$

and observe that (RGP) is equivalent to $\alpha_S(i,j) \geq \min[\alpha_S(i,k), \alpha_S(k,j)]$. This in particular says that $d_S(i,j) = \alpha_S(i,j)^{-1}$ is a metric satisfying the following stronger version of the triangle inequality: $d_S(i,j) \leq \max[d_S(i,k), d_S(k,j)]$. Such metrics are called *ultrametrics* and have the interesting properties that all triangles are isosceles. This translates back to α_S as saying that given $\{i,j,k\}$, then up to renaming we have:

$$\alpha_S(i,k) = \alpha_S(k,j) \leq \alpha_S(i,j) \tag{Iso}$$

We observe one interesting non-trivial and useful consequence of the isosceles triangle property, which will be useful later:

LEMMA 4.3. *Given a gross substitute valuation $v : 2^{[n]} \to \mathbb{R}$, $S \subseteq [n]$ and $i_1, i_2, j_1, j_2 \notin S$, then:*

$$v(i_1,i_2|S) + v(j_1,j_2|S) \leq \max[v(i_1,j_2|S) + v(j_1,i_2|S), v(i_1,j_1|S) + v(i_2,j_2|S)]$$

Proof. Let $P = \{i_1,i_2,j_1,j_2\}$ and $M = \min_{\{t_1,t_2\}\in P} \alpha_S(t_1,t_2)$. Define the *length* of the edge between (t_1,t_2) as $\alpha_S(t_1,t_2)$. By property (Iso), every triangle is isosceles with the smaller edge appearing at least twice. So, if we look the graph of the edges of minimal length between P, then either one of two things happen: (i) there is a cyle, i.e., there are $\alpha_S(x_1,x_2) = \alpha_S(x_2,x_3) = \alpha_S(x_3,x_4) = \alpha_S(x_4,x_1) = M$ where $\{x_1,x_2,x_3,x_4\} = \{i_1,i_2,j_1,j_2\}$ or (ii) there is a star, i.e., $\alpha_S(x_1,x_2) = \alpha_S(x_2,x_3) = \alpha_S(x_2,x_4) = M$. In order to see that, let x_1, x_2 be nodes in P such that $\alpha_S(x_1,x_2) = M$. Let x_3 be some other node, so the triangle x_1,x_2,x_3 must have two sides of length M. Up to renaming, $\alpha_S(x_1,x_2) = \alpha_S(x_2,x_3) = M$. If $\alpha_S(x_2,x_4) = M$ we are in case (ii). If $\alpha_S(x_2,x_4) > M$ then by property (Iso) applied on the triangles x_1,x_2,x_4 and x_2,x_3,x_4, we must have $\alpha_S(x_1,x_4) = \alpha_S(x_3,x_4) = M$, in which case we are in case (i).

Now, proving the statement of the lemma in each of the two cases is simple: if we are in case (i), then we can assume (swapping the names of i_1, i_2 if necessary) that $\alpha_S(i_1,j_1) = \alpha_S(i_2,j_2) = M$, so: $\alpha_S(i_1,j_1) + \alpha_S(i_2,j_2) = 2M \leq \alpha_S(i_1,i_2) + \alpha_S(j_1,j_2)$, which is equivalent to the statement in the lemma. In case (ii), say i_1 is the center of the star, i.e., $\alpha_S(i_1,x) = M$ for all $x \in \{i_2,j_1,j_2\}$. Now: $\alpha_S(j_1,j_2) \geq \min[\alpha_S(i_2,j_1), \alpha_S(i_2,j_2)]$. Swapping the names of j_1 and j_2 if necessary, we can assume that: $\alpha_S(j_1,j_2) \geq \alpha_S(i_2,j_1)$. That together with $\alpha_S(i_1,i_2) = M = \alpha_S(i_1,j_2)$, we get again $\alpha_S(i_1,j_1) + \alpha_S(i_2,j_2) \leq \alpha_S(i_1,i_2) + \alpha_S(j_1,j_2)$ which is equivalent to the statement in the lemma. □

5. Well Layered and Matroidal Maps. The final step towards proving the equivalence of definitions 3.1 and 1.3 is the concept of *well layered maps* introduced by Dress and Terhalle [7] – in which the authors characterize the set functions $v : 2^{[n]} \to \mathbb{R}$ for which greedy algorithms are optimal.

DEFINITION 5.1 (well-layered map). *A function $v : 2^{[n]} \to \mathbb{R}$ is called well-layered iff for each $p \in \mathbb{R}^n$ the sets $S_0, S_1, S_2, \ldots$ obtained by the greedy algorithm (i.e., $S_0 = \emptyset$ and $S_i = S_{i-1} \cup \{x_i\}$ where $x_i \in argmax_{x \in [n] \setminus S_{i-1}} v_p(x|S_{i-1})$) are such that $v_p(S_i) = \max\{v_p(S); |S| = i\}$.*

THEOREM 5.2 (Dress and Terhalle [7]). *A map $v : 2^{[n]} \to \mathbb{R}$ is well-layered iff for any triple of disjoint sets $S, \{i\}, T$ with $|T| \geq 2$,*

$$v(i|S) + v(T|S) \leq \max_{j \in T} v(j|S) + v(T \cup i \setminus j|S) \tag{WL}$$

Before we proceed to the proof of Theorem 5.2 it is useful to notice its relation with the condition of Reijnierse et al:

LEMMA 5.3. *Conditions (RGP) and (WL) are equivalent.*

Proof. One readily recognizes condition (RGP) to be a special case of (WL) with $|T| = 2$. For the other direction, we show by induction on $|T|$ that (RGP) implies (WL). For $|T| = 2$, this is trivial. Now, suppose we proved it for $|T| = t - 1$. Given $S, \{i\}, T$ with $|T| = t$, choose $k \in T$ minimizing $\alpha_S(i, k)$. Then by the induction hypothesis applied to $S \cup k, \{i\}, T \setminus k$, we have that there is $j \in T \setminus k$ such that:

$$v(i|S \cup k) + v(T \setminus k|S \cup k) \leq v(j|S \cup k) + v(T \cup i \setminus j, k|S \cup k)$$

which can be re-written as:

$$v(i, k|S) + v(T|S) \leq v(j, k|S) + v(T \cup i \setminus j|S)$$

now notice that by the choice of k, it must be the case that for all j, $\alpha_S(i, k) \leq \alpha_S(k, j)$, since the smaller α_S-value in the triangle i, j, k appears twice and $\alpha_S(i, k) \leq \alpha_S(i, j)$. Now, this means that $v(i|S) - v(i, k|S) \leq v(k|S) - v(j, k|S)$. Summing with the previous inequality gives us condition (WL) for $|T| = t$. □

Proof of Theorem 5.2. First, assume that the condition (WL) holds and let's prove that $S_t \in \text{argmax}_{S;|S|=t} v_p(S)$ by induction on t. The case $t = 1$ is trivial. Assume we proved for $t - 1$ and assume there is S' with $|S'| = t$ and $v_p(S') > v_p(S_t)$. Choose such set maximizing k such that $\{x_1, \ldots, x_k\} \subseteq S'$. Since $|S'| = |S_t| = t$, clearly $k < t$. Now, applying (WL) for $\{x_1, \ldots, x_k\}, x_{k+1}, T' = S' \setminus \{x_1, \ldots, x_k\}$ we get that there is $j \in T'$ such that:

$$v(x_{k+1}|x_1, \ldots, x_k) + v(T'|x_1, \ldots, x_k) \leq v(j|x_1, \ldots, x_k) + v(T' \cup x_{k+1} \setminus j|x_1, \ldots, x_k)$$

and since $v(x_{k+1}|x_1, \ldots, x_k) \geq v(j|x_1, \ldots, x_k)$ by the greedy rule, we have that $v(S') \leq v(S' \cup x_{k+1} \setminus j)$ contradicting the minimality of k.

For the other direction, assume that (WL) is violated. Since (WL) is equivalent to (RGP), (WL) must be violated by some $|T| = \{j, k\}$. So assume $S, i, \{j, k\}$ for which (WL) is not valid. Now, define prices such that $p_t = -\infty$ for $t \in S$, $p_t = \infty$ for $t \notin S \cup \{i, j, k\}$, $p_i = v(i|S) - \epsilon$, $p_j = v(j|S)$ and $p_k = v(k|S)$. The greedy algorithm will first pick all the elements in S, then i. Now, observe that for $t = |S| + 2$ the optimal set is either $S \cup \{i, j\}$, or $S \cup \{j, k\}$ or $S \cup \{i, k\}$. The fact that (WL) is violated implies that $v(i|S) + v(j, k|S) > v(j|S) + v(i, k|S)$. Substituting $v(i|S)$ and $v(j|S)$ by

the prices, we get that (for sufficiently small ϵ) $v(j,k|S)-p_j-p_k > v(i,k|S)-p_i-p_k$, i.e., $S\cup\{j,k\}$ is strictly preferable then $S\cup\{i,k\}$. The exact same argument works swapping j and k, so the only set of size $|S|+2$ maximizing v_p is $S\cup\{j,k\}$. Therefore v can't be a well-layered map, since the greedy algorithm picked i in step $|S|+1$. This finishes the proof of Theorem 5.2. □

The concept of well layered maps guarantees that the greedy algorithm will find the optimal set of each cardinality. In order to guarantee that the greedy algorithm as described in Section 3 will find the optimal, we need to guarantee that once a layer doesn't improve over the previous, we can stop. Dress and Terhalle [6] observe that in order for this to happen, it is necessary and sufficient that the valuation is both well-layered and submodular. They call such functions *matroidal maps*.

THEOREM 5.4 (Dress and Terhalle [6]). *A valuation function satisfied Definition 3.1 iff it is well-layered and submodular.*

Proof. Submodularity guarantees that the sets S_t will be such that $v_p(S_{t+1}) - v_p(S_t) \leq v_p(S_t\cup x_{t+1}\setminus x_t) - v_p(S_{t-1}) \leq v_p(S_t) - v_p(S_{t-1})$. So: $v_p(S_t) \geq \frac{1}{2}[v_p(S_{t-1}) + v_p(S_{t+1})]$. This guarantees that $t\mapsto v_p(S_t)$ is concave. For the converse, if a function is not submodular, then there exist i,j,S such that $v(i,j|S) > v(i|S)+v(j|S)$. So one can set prices $p_t=-\infty$ for $t\in S$, $p_t=\infty$ for $t\notin S\cup\{i,j\}$, $p_i = v(i|S)+\epsilon$ and $p_j = v(j|S)+\epsilon$ for some small ϵ. For such prices the greedy algorithm will terminate on S, while the optimum is $S\cup\{i,j\}$. □

Theorem 5.4 establishes our main claim that Definitions 3.1 and 1.3 are equivalent. In particular, for any gross substitute valuation function, a set $S\in D(v;p)$ can be found using the greedy algorithm (Algorithm 2). The following remark states that the greedy algorithms finds not only one, but all demanded sets. This fact is somehow implicit in the proof of Theorem 5.4 but we state and prove it again for completeness:

REMARK 5.5. *Given a gross substitute valuation function v, a price vector p and a demanded set $S\in D(v;p)$, then there is a tie breaking rule*[4] *for the greedy algorithm that produces the set S. Formally, if $x_1,x_2,\ldots,x_k$ is an ordering of elements in S such that $x_i \in argmax_{y\in S\setminus\{x_1,\ldots,x_{i-1}\}} v(y|x_1,\ldots,x_{i-1})$, then x_i is also in $argmax_{y\in[n]\setminus\{x_1,\ldots,x_{i-1}\}} v(y|x_1,\ldots,x_{i-1})$.*

Proof. This fact follows directly from the (WL) condition. Assume that there is i such that for some $v_p(y|x_1,\ldots,x_{i-1}) > v_p(x_i|x_1,\ldots,x_{i-1})$. Since $x_1,\ldots,x_k$ are in greedy order, it must be that $y\notin S$. By applying (WL) with sets $X_{i-1} = \{x_1,\ldots,x_{i-1}\},\{y\},\{x_i,\ldots,x_k\}$, we get that there is x_t for $t\geq i$ such that:

$$v_p(y|X_{i-1}) + v_p(x_i,\ldots,x_k|X_{i-1}) \leq v_p(x_t|X_{i-1}) + v_p(\{y,x_i,\ldots,x_k\}\setminus x_t|X_{i-1})$$

Since $v_p(y|X_{i-1}) > v_p(x_i|X_{i-1})$, we get $v_p(S) < v_p(S\cup y\setminus x_t)$, which contradicts that $S\in D(v;p)$. □

We would like to finish by pointing out that many combinatorial structures such as matroids, polymatroids, valuated matroids [8], among others, can be defined as

[4] formally a *tie breaking rule* is a funtion b that associates for each pair of disjoint sets S,T an element from T. The greedy algorithm with tie-breaking rule b, whenever choosing an element in $T=\text{argmax}_{i\notin S} v(i|S)$, it adds $b(S,T)\in T$.

the class of objects for which a certain problem admits a greedy solution. For a more extensive exposition on such combinatorial structures we refer to the classical text of Korte, Lovász and Schrader on *greedoids* [18]. Similar arguments can be used to argue about local search:

THEOREM 5.6. *A valuation function satisfied Definition 3.3 iff it is well-layered and submodular.*

Proof. First we argue that if a valuation v is well-layered and submodular, then local search can't get stuck in a local maximum for any price p. Let $S^* \in \text{argmax}_{S\subseteq[n]} v_p(S)$ and $S \subseteq [n]$ be such that $v_p(S) < v_p(S^*)$. Then we want to argue that there is $S' \in \mathcal{N}$ where $\mathcal{N}$ is the neighborhood of S as in the local search procedure in Section 3. First observe that if v is well-layered and submodular, then v_p has also those two properties.

We consider three cases:

Case (i) $S \subseteq S^*$. Notice that $0 < v_p(S^* \setminus S|S) \leq \sum_{i\in S^*\setminus S} v_p(i|S)$, where the last inequality follows from submodularity. Therefore, there is some i for which $v_p(i|S) > 0$, then we can take $S' = S \cup \{i\}$.

Case (ii) $S^* \subseteq S$. Let $S \setminus S^* = \{i_1, \ldots, i_k\}$ For this case, $0 > v_p(S \setminus S^*|S^*) = \sum_{j=1}^{k} v_p(i_j|S \cup \{i_1, \ldots, i_{j-1}\}) \geq \sum_{j=1}^{k} v_p(i_j|S \setminus i_j)$. So there must be $i \in S \setminus S^*$ such that $v_p(i|S \setminus i) < 0$, then we can take $S' = S \setminus i$.

Case (iii) if neither $S \subseteq S^*$ nor $S^* \subseteq S$, let i be the element in $(S \setminus S^*) \cup (S^* \setminus S)$ maximizing $v_p(i|S \cap S^*)$. Now, we consider two possibilities: (iii-a) If $i \in S^*$, we use condition (WL) with $S \cap S^*, i, S \setminus S^*$. This gives us $j \in S \setminus S^*$ such that:

$$v_p(i|S \cap S^*) + v_p(S \setminus S^*|S \cap S^*) \leq v_p(j|S \cap S^*) + v_p((S \setminus S^*) \cup i \setminus j|S \cap S^*)$$

By the choice of i, we know that $v_p(j|S \cap S^*) \leq v_p(i|S \cap S^*)$. If this inequality holds strictly, then, $v_p(S \setminus S^*|S \cap S^*) < v_p((S \setminus S^*) \cup i \setminus j|S \cap S^*)$ and therefore $v_p(S) \leq v_p(S \cup i \setminus j)$. If on the other hand $v_p(j|S \cap S^*) = v_p(i|S \cap S^*)$ then we can swap i and j and consider the subcase where $i \in S$. (iii-b) If $i \in S$, we use condition (WL) with $S \cap S^*, i, S^* \setminus S$. Doing as above, we find $j \in S^* \setminus S$ such that $v_p(S^*) \leq v_p(S^* \cup i \setminus j)$, which holds with equality since S^* is optimal. This way we obtain an optimal set closer to S. Then we can repeat the above procedure with $S^* \cup i \setminus j$ instead of S^* a finite number of times until we reach some set $S' \in \mathcal{N}$ or we reach one of the previous cases.

For the converse direction, we want to show that if a valuation is not well-layered or not submodular, local search can get stuck in suboptimal local minima. For this, we can use the same examples used to show that in such case the greedy algorithm can be suboptimal. □

We end this section with Figure 5.1 summarizing the equivalences proved so far. The dashed arrow corresponds to a claim proved in [29] but not fully proved in this survey. To make the survey self-contained, we give a direct proof that the concept of matroidal maps (Definition 3.1) implies the original definition of gross substitutes due to Kelso and Crawford (Definition 1.3):

THEOREM 5.7. *Every matroidal map satisfied the gross substitutes condition.*

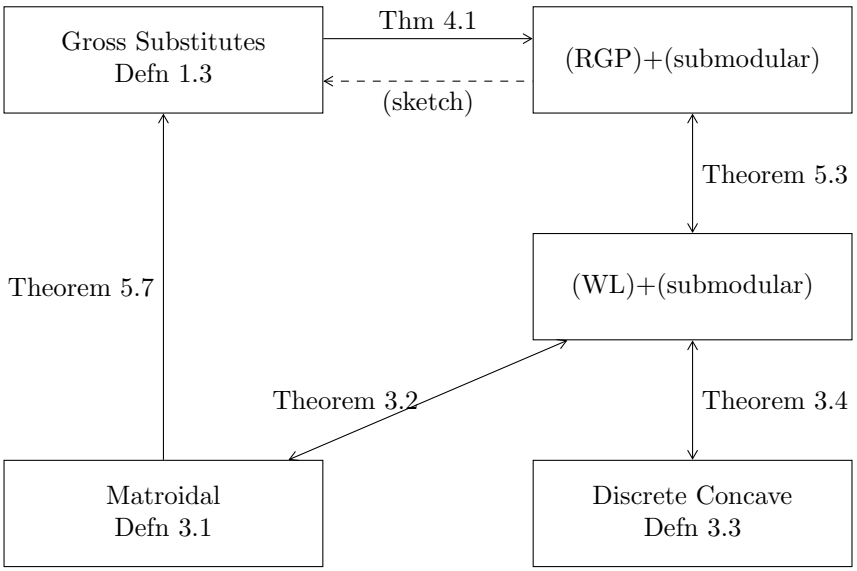

FIG. 5.1. *Summary of equivalences proved so far*

Proof. Consider a matroidal map v, a price vector p and a set $S \in D(v;p)$. In order to show that the conditions in Definition 1.3 hold, we only need to show for a price vector p' such that $p'_i > p_i$ for some i and $p'_j = p_j$ for all $j \neq i$, since we can increase prices one at a time. Let S' be a set in $\text{argmax}_{S' \in [n] \setminus i} v_p(S')$. Clearly, if $p'_i \leq p_i + v_p(S) - v_p(S')$, then $S \in D(v;p')$ and we are done. Now, consider the case where $p'_i > p_i + v_p(S) - v_p(S')$. For this, first define $\tilde{p}$ such that $\tilde{p}_i = p_i + v_p(S) - v_p(S')$ and $\tilde{p}_j = p_j$ for all $j \neq i$. Clearly $S \in D(v;\tilde{p})$. By Remark 5.5, there is a tie breaking rule for which the greedy algorithm picks S. Since $v_{\tilde{p}}(i|S \setminus i) = 0$, we can assume that i is the last element picked under this tie breaking rule. One can use the same tie breaking rule to pick under price vector p'. It is straightforward to see that the greedy algorithm behaves the same way as in $\tilde{p}$ until i. Therefore it will choose a set containing $S \setminus i$. □

6. Duality theorem for gross substitutes. The duality between gross substitutes and submodular functions was observed in many places, as in Fujishige and Yang [9] and Murota [23], Gul and Stacchetti [12] and Ausubel and Milgrom [1]. Given a valuation function v, they consider the utility function $u : \mathbb{R}^n \to \mathbb{R}$ which maps a set of prices $p \in \mathbb{R}^n$ to the optimal utility that can be obtained under such prices $u(p) = \max_S v_p(S)$. They relate it to the concept of $\mathbb{R}^n$-valued submodular function, which are functions $f : \mathbb{R}^n \to \mathbb{R}$ such that for any $x, y \in \mathbb{R}^n$, $f(x \vee y) + f(x \wedge y) \leq f(x) + f(y)$, where $\vee$ and $\wedge$ are the componentwise maximum and minimum respectively. Analogously to submodular set functions, this is equivalent to $f(x + \delta_i \cdot e^i + \delta_j \cdot e^j) - f(x + \delta_i \cdot e^i) \leq f(x + \delta_j \cdot e^j) - f(x)$ for any $\delta_i, \delta_j > 0$ and $i \neq j$, where e_i is the i-th coordinate vector.

THEOREM 6.1 (duality, Ausubel and Milgrom [1]). *A valuation function $v : 2^{[n]} \to \mathbb{R}$ has the gross substitutes property iff its associate utility $u(p) = \max_S v_p(S)$ is an $\mathbb{R}^n$-valued submodular function.*

Proof. Since u is continuous, it is enough to prove for almost all p, δ_i, δ_j and then we can extend to all by continuity. Define $\Gamma = \cup_{S,T\subseteq[n]}\{p \in \mathbb{R}^n; v_p(S) = v_p(T)\}$. Then Γ is a measure zero subset of $\mathbb{R}^n$, since it is a finite collection of hyperplanes. For $p \notin \Gamma$, $|D(v,p)| = 1$. For $p \notin \Gamma$, denote by $D(v,p)$ the unique set demanded at those prices. Given such p, there is ball around p such that for all price vectors, the demand set is the same, so $u(p) = v(D(v,p)) - p(D(v,p))$ and therefore, $\frac{d}{dp_j}u(p) = -\mathbb{1}\{j \in D(v,p)\}$. Now, notice that given p, δ_i, δ_j:

$$[u(p + \delta_i \cdot e^i + \delta_j \cdot e^j) - u(p + \delta_i \cdot e^i)] - [u(p + \delta_j \cdot e^j) - u(p)] =$$
$$\int_0^{\delta_j} \frac{d}{dp_j}u(p + \delta_i \cdot e^i + z \cdot e^j) - \frac{d}{dp_j}u(p + z \cdot e^j)dz =$$
$$\int_0^{\delta_j} -\mathbb{1}\{j \in D(v, p + \delta_i \cdot e^i + z \cdot e^j)\} + \mathbb{1}\{j \in D(v, p + z \cdot e^j)\}dz \leq 0$$

since the increase in p_i can't remove j from the demand set by gross substitutability. The converse can be proved by the same argument backwards.
□

A different view of the same duality can be obtained by the characterization of demand sets as basis of a matroid:

THEOREM 6.2 (duality, Gul and Stacchetti [12]). *A valuation function $v : 2^{[n]} \to \mathbb{R}$ has the gross substitutes property then for any price $p \in \mathbb{R}^n$, the set*

$$D^*(v,p) = \{S \in D(v,p); |S| \leq |T|, \forall T \in D(v,p)\}$$

of the demanded sets of minimum size, form the set of basis of a matroid.

One characterization of basis-set $\mathcal{B} \subset 2^{[n]}$ of matroids is via the exchange property: for any $S, T \in \mathcal{B}$ and $s \in S \setminus T$, there is $t \in T \setminus S$ such that $S \cup t \setminus s \in \mathcal{B}$. We notice that we proved exactly this fact in case (iii) of the proof of Theorem 5.6.

7. Connection to Discrete Convex Analysis and Valuated Matroids. Fujishige and Yang [9] showed a powerful connection between gross substitute valuations and the concept of $M^\natural$-concave functions in Discrete Convex Analysis. Discrete Convex Analysis is a theory developed by Murota [23] that defines a very general class of functions $f : \mathbb{Z}^n \to \mathbb{R}$ on the integral lattice for which it is possible to prove strong duality theorems. Such theorems enable the design of efficient greedy and flow-like algorithmic solutions for various discrete optimization problems involving such functions.

Murota and Shioura [26] define $M^\natural$-concave functions based on the concept of M-concavity of Murota [23]. They originally define $M^\natural$-concave functions on the integral lattice $\mathbb{Z}^n$, but for the purposes of this survey, we will consider their restriction to $\{0,1\}^n$:

DEFINITION 7.1 ($M^\natural$-concave functions, Murota and Shioura [26]). *A function $v : 2^{[n]} \to \mathbb{R}$ is $M^\natural$-concave if for all $S, T \subseteq [n]$ and $s \in S \setminus T$,*

$$v(S) + v(T) \leq \max\left[v(S \setminus s) + v(T \cup s), \max_{t \in T \setminus S} v(S \cup t \setminus s) + v(T \cup s \setminus t)\right] \qquad (M^\natural)$$

THEOREM 7.2 (Fujishige and Yang [9]). *A function $v : 2^{[n]} \to \mathbb{R}$ has the gross substitutes property iff it is $M^\natural$-concave.*

Proof. The fact that $M^\natural$-concavity implies gross substitutability is easy to see, since taking $T \subseteq S \setminus s$ we recover submodularity: $v(S) + v(T) \leq v(S \setminus s) + v(T \cup s)$ which can be rewritten as $v(s|S \setminus s) \leq v(s|T)$. Taking $|S \setminus T| = 1$ and $|T \setminus S| \geq 2$, we recover (WL). Since by submodularity $v(S) + v(T) \geq v(S \setminus s) + v(T \cup s)$, so if $v(S)+v(T) \leq v(S\setminus s)+v(T\cup s)$, then $v(U\cup s) = v(U)+v(s|T)$ for any $S \subseteq U \subseteq T$, making (WL) hold for any $t \in T\setminus S$. On the other hand, if $v(S)+v(T) > v(S\setminus s)+v(T\cup s)$, then: $v(S) + v(T) \leq \max_{t\in T\setminus S} v(S \cup t \setminus s) + v(T \cup s \setminus t)$, which is exactly (WL).

For the other direction, assume that v is gross substitutes and we want to show is satisfied ($M^\natural$). First, consider the following transformation: given $v : 2^{[n]} \to \mathbb{R}$, define another valuation function on $2n$ items by adding n dummy items: $\omega : 2^{[2n]} \to \mathbb{R}$, $\omega(S) = v(S \cap [n])$. It is straightforward to check that if v is gross substitutes then so does ω. Now we define the condition (M) on ω as follows: for all sets $S, T \subseteq [2n]$ with $|S| = |T|$ and $s \in S \setminus T$:

$$\omega(S) + \omega(T) \leq \max_{t\in T\setminus S} \omega(S \cup t \setminus s) + \omega(T \cup s \setminus t) \tag{M}$$

It is simple to see that if ω satisfied (M) for all sets of equal cardinality, then v satisfied ($M^\natural$), since any pair of sets $S, T \subseteq [n]$ map to equal cardinality sets $S', T' \subseteq [2n]$ by padding the smaller set with dummy elements. Notice that (M) on ω implies ($M^\natural$) on v, since the term $v(S \setminus s) + v(T \cup s)$ accounts for the possibility that $t \in T \setminus S$ in (M) is a dummy item of $[2n]$.

So, we only need to prove that if ω is a gross substitutes valuation, then it also satisfied (M). For $|S \setminus T| = |T \setminus S| = 1$, the property in trivial. For $|S \setminus T| = |T \setminus S| = 2$, this follows directly from Lemma 4.3. Now we prove by induction on $k = |S \setminus T| = |T \setminus S|$. Fix some arbitrary $\tilde{s} \in S \setminus (T \cup s)$ and find $\tilde{t} \in T \setminus S$ maximizing $\omega(T\cup\tilde{s}\setminus\tilde{t})-\omega(S\cup\tilde{t}\setminus s)$. Now, apply induction on the sets S and $T\cup\tilde{s}\setminus\tilde{t}$. We get that there is $t \in T \setminus (S \cup \tilde{t})$ such that:

$$\omega(S) + \omega(T \cup \tilde{s} \setminus \tilde{t}) \leq \omega(S \cup t \setminus s) + \omega(T \cup \{s, \tilde{s}\} \setminus \{t, \tilde{t}\})$$

By the case with $k = 2$ with sets T and $T \cup \{s, \tilde{s}\} \setminus \{t, \tilde{t}\}$, we know that:

$$\omega(T)+\omega(T\cup\{s,\tilde{s}\}\setminus\{t,\tilde{t}\}) \leq \max[\omega(T\cup s\setminus t)+\omega(T\cup\tilde{s}\setminus\tilde{t}),\omega(T\cup\tilde{s}\setminus t)+\omega(T\cup s\setminus\tilde{t})]$$

If the maximum corresponds to the first expression, this together with the previous inequality, gives us exactly what we want to prove, i.e., $\omega(S) + \omega(T) \leq \omega(S \cup t \setminus s) + \omega(T\cup s\setminus t)$, which corresponds to condition (M). If the maximum is the second expression we use the choice of $\tilde{t}$ to see that: $\omega(T\cup\tilde{s}\setminus\tilde{t})-\omega(S\cup\tilde{t}\setminus s) \geq \omega(T\cup\tilde{s}\setminus t)-\omega(S\cup t\setminus s)$. This together with the previous inequalities also leads to condition (M). □

Valuated Matroids. The characterization of gross substitutability by the ($M^\natural$) also connects it to the concept of *valuated matroids*, due to Dress and Wenzel [8]:

DEFINITION 7.3. *Let $\binom{[n]}{k} = \{S \subseteq [n]; |S| = k\}$. We say that a map $\omega : \binom{[n]}{k} \to \mathbb{R}$ is a* valuated matroid *if is satisfied the following version of the exchange property: given $S, T \in \binom{[n]}{k}$ and $s \in S \setminus T$, there exists $t \in T \setminus S$ such that:*

$$\omega(S) + \omega(T) \leq \omega(S \cup t \setminus s) + \omega(T \cup s \setminus t)$$

In particular, Theorem 7.2 together with the discussion in its proof imply that:

LEMMA 7.4. *A valuation $v : 2^{[n]} \to \mathbb{R}$ satisfies the gross substitutes property iff the map $\omega : \binom{[2n]}{n} \to \mathbb{R}$ defined by $\omega(S) = v(S \cap [n])$ is a valuated matroid. Also, if v satisfies the gross substitutes property then for every $k \leq n$, the restriction of v to $\binom{[n]}{k}$ is a valuated matroid. In other words, given two sets of equal cardinality and a gross substitutes valuation, then (M) is satisfied.*

8. Convolution operation. As we saw in Section 2, one *cannot* build gross substitute functions by taking linear combinations of simpler gross substitute, since gross substitutability is not closed under addition. However, this class is closed under a different operation, called convolution.

THEOREM 8.1 (Lehmann, Lehmann and Nisan [21] and Murota [23]). *Given two valuation functions v^1, v^2 satisfying the gross substitutes property then the valuation function $v = v^1 * v^2$ also satisfied the gross substitutes property, where:*

$$v^1 * v^2(S) = \max_{S_1 \subseteq S} v^1(S_1) + v^2(S \setminus S_1)$$

Proof. We will give an algorithmic proof of the previous theorem based on a couple of observations about the Walrasian tatônnement procedure. First note that we can find $S \in D(v^1 * v^2, p)$ by finding a Walrasian equilibrium in an economy with items $[n]$ and three players with valuations v^1, v^2, u where $u(S) = \sum_{j \in S} p_j$. Let S_1, S_2, U be the partition of the items induced by such equilibrium. Then, this is the partition maximizing $v_1(S_1) + v_2(S_2) + p(U) = p([n]) + [v_1(S_1) + v_2(S_2) - p(S_1 \cup S_2)]$. In particular, S_1, S_2 must be the optimal partition of $S = S_1 \cup S_2$ among v^1, v^2, therefore, S maximizes $(v^1 * v^2)(S) - p(S)$ and therefore $S \in D(v^1 * v^2, p)$.

Second observe that for gross substitute valuations the Walrasian tatônnement procedure (Algorithm 1) always outputs a partition of the goods if we are careful to always select $X_i \in D(v^i, p^i)$ such that $S_i \subseteq X_i$. This can be easily implemented by computing X_i via the greedy algorithm (Algorithm 2) initialized with $X_i = S_i$. Moreover, the partition is such that $\sum_i v^i(S_i) \geq \sum_i v^i(S_i^*) + \delta n$. For rational valuations, we can rescale them such that $v^i(S_i)$ are integers. In such case, taking $\delta < \frac{1}{n}$ guarantees that Walrasian tatônnement outputs the optimal allocation.

Finally, in the description of Algorithm 1 we initialized the prices as zero and the allocations such that agent 1 initially has all the goods. Notice that it enough to initialize with a price $p \in \mathbb{R}^n_+$ and a partition $S_1, \ldots, S_n$ such that there is $X_i \in D(v^i, p)$ such that $S_i \subseteq X_i$.

Those observations together can be used to give an elementary proof of Theorem 8.1. We show that $v^1 * v^2$ satisfy the Definition 1.3. Let $S \in D(v^1 * v^2, p)$ and $(v^1 * v^2)(S) = v^1(S_1) + v^2(S_2)$. Consider a Walrasian equilibrium in the economy formed by v^1, v^2, u. Let q be the price vector in such equilibrium. By the Second Welfare Theorem, we take the allocation in equilibrium as $S_1, S_2, U = [n] \setminus (S_1 \cup S_2)$. Let p' be a price vector with $p \leq p'$. We want to show that there is a set $X \in D(v^1 * v^2, p')$ such that $S \cap \{j; p_j = p'_j\} \subseteq X$.

For that, define $u'(S) = \sum_{j \in S} p'_j$ and consider the Walrasian tatônnement procedure for the economy defined by v^1, v^2, u' . Initialize such procedure with allocation S_1, S_2, U and price vector q. This is a valid initialization, since $S_i \in D(v^i, q)$ and also,

$U \subseteq \{j; q_j \leq p'_j\} \in D(u', q)$. Now, let S'_1, S'_2, U' be the final outcome of the Walrasian tatônnement procedure. Observe that if $j \in S_1 \cup S_2$, then $q_j \geq p_j$, otherwise q wouldn't be Walrasian for v^1, v^2, u. Now, if $p'_j = p_j$, then such item couldn't have been acquired by u', since it would never be in his demand for such price. Therefore $j \in S'_1 \cup S'_2$. Hence, $(S_1 \cup S_2) \cap \{j; p_j = p'_j\} \subseteq S'_1 \cup S'_2$. □

A corollary of Theorem 8.1 is that assignment valuations are gross substitutes: it is straightforward from the definition that an assignment valuation function can be written as a convolution of unit-demand functions, one for each right-side node in the bipartite graph.

9. Computing Walrasian Prices for gross substitutes. The problem of computing a Walrasian equilibrium of an economy consisting of n items and m agents with gross substitutes valuations $v^1, \ldots, v^m$ has two components: the first is called the *welfare problem*, which consists in finding a partition $S_1, \ldots, S_m$ maximizing $\sum_i v^i(S_i)$. The second is the computation of *Walrasian prices*. There are various approaches for those problems: the perhaps more classical line of approach is to use variations of the tatônnement procedure. Nisan and Segal [27] propose a solution that explores properties of gross substitutes to build a suitable linear program. Finally, Murota [24, 25] gives a strongly polynomial time algorithm for this problem based on a cycle-cancelling approach.

We start by discussing how to obtain Walrasian prices from a solution to the welfare problem. The first method is based on an idea by Gul and Stachetti [11]:

LEMMA 9.1 (Gul and Stachetti [11]). *Let W be the optimal welfare of an economy with a set $[n]$ of items and agents with gross substitute valuations $v^1, \ldots, v^m$. Also, let W_{-j} be the welfare with the economy with the same agents and items $[n] \setminus j$. Then the price vector p with $p_j = W - W_{-j}$ is a vector of Walrasian prices for the original economy.*

The method proposed by Gul and Stachetti to compute Walrasian prices needs access to the optimal allocation for $n+1$ economies: the original one and the economy after each good is removed. An alternative approach is given by Murota [24]. First, consider the following definition:

DEFINITION 9.2 (Exchange graph). *Let $S_1, \ldots, S_m$ be an arbitrary allocation of a set $[n]$ of items to agents with gross substitute valuations $v^1, \ldots, v^m$. For convenience, we consider m additional dummy $d_1, \ldots, d_m$ and extend the valuations to this set in such a way that for each set S, $v^i(S) = v^i(S \cap [n])$. Also, let $S'_i = S_i \cup d_i$.*

The exchange graph is defined as a directed weighted graph with nodes $[n+m] = [n] \cup \{d_1, \ldots, d_m\}$ and weighted directed edges

$$(j, k) \text{ with weight } w_{jk} = -v^i(S_i \cup k \setminus j) + v^i(S_i) \text{ for } j \in S'_i \text{ and } k \notin S'_i$$

LEMMA 9.3 (Murota [24]). *If the allocation is optimal, the exchange graph has no negative cycles and therefore, the shortest-path distance is well defined. For each $i \in [n+m]$, let ϕ_i be the distance from dummy node d_1 to i. Then $\phi_i \leq 0$ and the*

vector $p_i = -\phi_i$ is a vector of Walrasian prices.

Proof. First assume that the graph has no negative cycles. In such case, the concept of distance is well defined. Given a pair of dummy nodes d_i, d_j, the weight of the arcs $w_{d_i,d_j} = w_{d_j,d_i} = 0$, then $\phi_{d_i} = 0$ for all dummy nodes. Also, for all items $k \in S_i$, the weight of the arc from a dummy node d_i to k is $w_{d_i,k} = -v^i(S_i \cup k) + v^i(S_i) \leq 0$, so $\phi_k \leq 0$ for all nodes k. This allows us to define prices as $p_k = -\phi_k$. Since ϕ is the shortest path distance, for all $j \in S_i', k \notin S_i'$: $\phi_k \leq \phi_j + w_{jk}$, which is equivalent to: $v^i(S_i) \geq v^i(S_i \cup k \setminus j) - p_k + p_j$. Since k and j are possibly dummy items, this also implies that: $v^i(S_i) \geq v^i(S_i \cup k) - p_k$ and $v^i(S_i) \geq v^i(S_i \setminus j) + p_j$. The last three inequalities show that S_i is a local optimal of the local search procedure (Algorithm 3), hence $S_i \in D(v^i, p)$ by Theorem 5.6. Therefore, p is a vector of Walrasian prices.

Now we argue that if the allocation is optimal, then there are no negative cycles. Given an optimal allocation, let p be a vector of Walrasian prices supporting this allocation. Now define $\phi_j = -p_j$ for all $j \in [n]$ and $\phi_d = 0$ for all dummy items d. By the same argument as above, the fact that p is a vector of Walrasian prices implies that: $w_{ij} \geq \phi_i - \phi_j$, therefore for every cycle $i_1, i_2, \ldots, i_k, i_1$ we have: $\sum_{t=1}^{k} w_{i_t i_{t+1}} \geq \sum_{t=1}^{j} \phi_{i_t} - \phi_{i_{t+1}} = 0$. ∎

The exchange graph has $O(n+m)$ nodes and $O((n+m)^2)$ edges. Checking if a graph has negative cycles can be done in time $O(N \cdot E)$ using the Bellman-Ford algorithm, where N is the number of nodes and E is the number of edges. This gives an $O((n+m)^3)$ algorithm to compute Walrasian prices for gross substitute valuations given the optimal allocation.

10. Welfare Problem for gross substitutes. Finally, we discuss algorithmic solutions to the welfare problem for gross substitute valuations. Before we start, we mention a couple of important special cases of this problem. If $v^i(S) = \max_{j \in S} w_{ij}$ for all $i \in [m]$, then this is the traditional *maximum weighted matching* problem. If v^i is the rank function of a matroid, then this corresponds to a special case of the *matroid intersection problem*. For example, the problem of deciding if a graph has k disjoint spanning trees naturally maps to the welfare problem with k agents where the items correspond to edges of the graph and valuation functions correspond to the rank function of the graphical matroid. We discuss three approaches for this problem: tatônnement, linear-programming and cycle cancelling. The first approach has a natural economic intuition but yields only an approximation scheme. The second approach produces an exact solution and runs in polynomial time. The third approach is purely combinatorial and yields a strongly polynomial time algorithm.

10.1. Algorithms via the tatônnement procedure. In Section 1, the Walrasian tatônnement procedure (Algorithm 1) was used as a proof device to show the existence of Walrasian equilibria for gross substitute valuations. In this section we discuss how to use it as an actual algorithm. We start by analyzing the running time of Algorithm 1 using the greedy algorithm (initialized with $X_i = S_i$) to compute the demand oracle. Then we discuss variants of the implementation.

We assume that $v^i(S)$ is an integer (rescaling the input, if necessary) and define $M = \max_{i \in [m]} v^i([n])$. We argued in Section 1 that each price can increase at most M/δ times. This gives a bound of nM/δ on the number of total price increases.

In what follows we argue that there are at most $m + nM/\delta$ executions of the *while* loop in Algorithm 1. Consider the following implementation of the while loop: start

with a queue containing all the agents $1, \ldots, m$. At each time, pop agent i from the the queue and compute $X_i \in D(v^i, p^i)$ with $S_i \subseteq X_i$. If $X_i \neq S_i$, execute the while loop and for each $k \neq i$ such that S_k changes during the while loop, add k to the queue if he is not already there.

Noticed at this point we removed i from the queue. After the execution of the while loop, we don't need to look at i again, *unless* S_i changes, i.e., unless some item j is taken away from i, since by the fact that valuations are gross substitutes and the prices only increase during the process, $S_i \in D(v^i, p^i)$ if the prices of items in S_i stay the same.

Each execution of the loop is dominated by the execution of the greedy demand oracle that takes $O(n^2)$ time. This gives us a total running time of $O(n^2(\frac{M}{\delta}n + m))$. This produces a partition $S_1, \ldots, S_m$ such that $\sum_i v^i(S_i) \geq \sum_i v^i(S_i^*) - \delta n$ as we argued in Section 1. Taking $\delta = \frac{1}{2n}$, gives us: $\sum_i v^i(S_i) \geq \sum_i v^i(S_i^*) - \frac{1}{2}$ and therefore $\sum_i v^i(S_i) = \sum_i v^i(S_i^*)$ since both are integers. The running time in this case is $O(n^2(Mn^2 + m))$.

The previous version runs in pseudo-polynomial time due to the linear dependency on M. This can be easily improved to a dependency on $\log(M)$ by updating the prices in a multiplicative fashion. Initialize all prices to zero and define the following price update rule: $U(0) = \delta$ and $U(p_j) = p_j(1 + \delta)$. So each price is in the set $\{0, \delta, \delta(1 + \delta), \delta(1 + \delta)^2, \ldots\}$. Now, we change Algorithm 1 in two ways: first we calculate p^i as $p^i_j = p_j$ if $j \in S_i$ and $p^i_j = U(p_j)$ otherwise. Also, when we update prices inside the while loop, we update p_j to $U(p_j)$. By the same argument as before, prices never rise past M, so there are at most $n \cdot \log_{1+\delta}(M)$ price updates. This produces a running time of $O(n^2m + \frac{1}{\delta}n^3 \log M)$. The solution produced is such that $v^i(S_i) - p(S^i) \geq v^i(T) - p(T) - \delta|T \setminus S_i| - \delta p(T \setminus S_i)$ for all $T \subseteq [n]$. Taking $T = S_i^*$ (the optimal partition) and summing for all i, we get: $(1 + \delta)\sum_i v^i(S_i) \geq \sum_i v^i(S_i^*) - \frac{n}{\delta}$.

Maximum matching. It is illuminating to look at the case of weighted maximum matching, i.e. $v^i(S) = \max_{j \in S} w_{ij}$. For this particular case, the Walrasian tatônnement procedure takes the form of the *auction method* from Bertsekas [2] and the ascending auction of Demange, Gale and Sotomayor [5]. It also closely resembles Kuhn's Hungarian Method [19]. For this particular case, the demand oracle can be computed in time $O(n)$, which gives us complexity $O(\frac{M}{\delta}n^2 + nm)$. Consider further the special case of unweighted maximum matching, where $n = m$, $w_{ij} \in \{0, 1\}$ and the graph has a perfect matching. Since $M = 1$, this gives an $(1-\delta)^{-1}$-approximation algorithm of running time $O(\frac{1}{\delta}n^2)$. Taking $\delta = \frac{1}{2}$ we get exactly the 2-approximation via the greedy algorithm for maximum matching. For $\delta = \frac{1}{n}$ we get an $O(n^3)$ exact algorithm. Taking $\delta = \frac{1}{\sqrt{n}}$ one gets a matching of size $n - \sqrt{n}$ in time, $O(n^2\sqrt{n})$. After more $\sqrt{n}$ iterations of an augmenting path algorithm, we are able to find the optimal matching with total running time $O(n^2\sqrt{n})$, which is the bound provided by the Hopcroft-Karp algorithm [14].

10.2. Linear Programming algorithms. The second approach, proposed by Nisan and Segal [27], is based on linear programming. They observe that the welfare problem can be cast as the following integer program:

$$
\begin{aligned}
W_{\text{IP}} = \max \sum_{i=1}^{m} \sum_{S \subseteq [n]} x_{iS} \cdot v^i(S) \qquad & \text{s.t.} \\
\sum_{S \ni j} \sum_{i} x_{iS} = 1, \quad & \forall j \in [n] \\
\sum_{S} x_{iS} = 1, \qquad & \forall i \in [m] \\
x_{iS} \in \{0,1\}, \qquad & \forall i \in [m], S \subseteq [n]
\end{aligned}
$$

Let W_{LP} correspond to the linear programming relaxation of the previous problem, i.e., to the program obtained by relaxing the last constraint to $0 \leq x_{iS} \leq 1$. Since it is a relaxation, $W_{\text{IP}} \leq W_{\text{LP}}$. Bikhchandani and Mamer [3] observe that when v^i are gross substitute valuations, this holds with equality, for the following reason: by the duality theorem in Linear Programming, W_{LP} corresponds to the solution of the following dual program:

$$
\begin{aligned}
W_{\text{LP}} = \min \sum_{i \in [m]} u_i + \sum_{j \in [n]} p_j \qquad & \text{s.t.} \\
u_i \geq v^i(S) - \sum_{j \in S} p_j, \quad & \forall i \in [m], S \subseteq [n] \\
p_j \geq 0, u_i \geq 0, \qquad & \forall i \in [m], j \in [n]
\end{aligned}
$$

An optimal solution to the integer program corresponds to the welfare of a Walrasian equilibrium $W_{\text{IP}} = \sum_i v^i(S_i)$. If p are the corresponding Walrasian prices and $u_i = v^i(S_i) - \sum_{j \in S_i} p_j$, (u, p) corresponds to a feasible solution to the dual. Therefore $W_{\text{LP}} \leq W_{\text{IP}}$.

Given this observation, Nisan and Segal propose solving the welfare problem by solving the dual linear program above using a separation based linear programming algorithm, such as the ellipsoid method. The program has $n + m$ variables but an exponential number of constraints. In order to solve it, we need to provide a separation oracle, i.e., an polynomial-time algorithm to decide, for each (u, p) if it is feasible and if not, produce a violated constraint. The problem that the separation oracle needs to solve is to decide for each agent i if $u_i \geq \max_S v^i(S) - \sum_{j \in S} p_j$. For gross substitute valuations, this can be easily solved by the greedy algorithm (Algorithm 2).

The algorithm described above computes the value $W^* = \sum_i v^i(S_i)$ of the optimal allocation. In order to compute the optimal allocation itself, one can proceed as follows: fix an arbitrary item j and for all agents i' compute the value of the optimal allocation, conditioned on agent i receiving j. In order words, compute the value of the optimal allocation of items $[n] \setminus j$ to agents with valuation $\tilde{v}^i = v^i$ for $i \neq i'$ and $\tilde{v}^{i'}(S) = v^{i'}(S|j)$ and let $W^*_{j \to i'}$ be the solution. Then there must be some agent i' for which $W^* = v^{i'}(j) + W^*_{j \to i'}$. Allocate j to this agent and recursively solve the allocation problem with on $[n] \setminus j$ for valuations $\tilde{v}$.

10.3. Cycle-canceling algorithms. Finally we describe a purely combinatorial approach proposed by Murota [24, 25] based on the Klein's cycle-cancelling technique [17] for the minimum-cost flow problem. This approach has the advantage that it

leads to a strongly polynomial time algorithm. The algorithm is presented in three parts: first we provide a generic cycle canceling algorithm that finishes in finite time but without additional running time guaranteed. Then, we show that a careful choice of cycles can lead to a (weakly) polynomial time algorithm, following an analysis by Goldberg and Tarjan [10]. Finally, an additional idea due to Zimmermann [33] leads to a strongly polynomial time algorithm.

10.3.1. Improving the allocation along negative cycles. Murota's optimality criteria (Lemma 9.3) states that an allocation $S_1, \ldots, S_m$ is suboptimal iff the exchange graph (Definition 9.2) contains a negative-weight cycle.

Let C be this negative weight cycle. First we note that an edge going out of S'_i (S_i augmented with a dummy item) corresponds to the exchange of a (possibly dummy) element $a^i \in S'_i$ by an element $b^i \notin S'_i$. The value of the edge corresponds to the change in value for i by replacing a^i by b^i, i.e., $w_{a^i b^i} = v^i(S_i) - v^i(S_i \cup b^i \setminus a^i)$.

This observation suggests the following approach to improve the allocation: perform the exchanges prescribed by such cycle, i.e., if $M^i = \{(a^i_1, b^i_1), \ldots (a^i_{k^i}, b^i_{k^i})\}$ is the set of edges in C going from S'_i, then update S_i to $S_i \cup \{b^i_1, \ldots, b^i_{k^i}\} \setminus \{a^i_1, \ldots, a^i_{k^i}\}$ (ignoring the dummy nodes). The change in welfare is given by $\sum_i v^i(S_i) - v^i(S_i \cup \{b^i_1, \ldots, b^i_{k^i}\} \setminus \{a^i_1, \ldots, a^i_{k^i}\})$ while the sum of weights of edges in the cycle is given by $\sum_i \sum_{(a,b)\in M^i} v^i(S_i) - v^i(S_i \cup b \setminus a)$. In general, those two quantities are different, so the fact that the cycle has negative weight is not enough to guarantee that performing the exchanges prescribed by it will result in an improvement in welfare.

Murota shows in [25] a sufficient condition for the total weight of the cycle to be equal to the change in welfare by performing the exchanges.

LEMMA 10.1 (Unique Minimum Weight Matching Condition). *Given a gross substitute valuation v, a set S, $A = \{a_1, \ldots, a_k\} \subseteq S$, $B = \{b_1, \ldots, b_k\} \subseteq [n] \setminus S$, consider the bipartite graph G with left nodes A, right nodes B and edge weights $w_{a_i b_j} = v(S) - v(S \cup b_j \setminus a_i)$. If $M = \{(a_1, b_1), \ldots, (a_k, b_k)\}$ is the unique minimum weight matching in the graph, then:*

$$v(S) - v(S \cup B \setminus A) = \sum_{j=1}^{k} v(S) - v(S \cup b_j \setminus a_j)$$

Proof. The proof of the lemma follows by induction on k. For $k = 1$, the theorem is trivial. Assume now it holds for $k-1$. First observe that: $v(S) - v(S \cup B \setminus A) = w_{a_k b_k} + v(S \cup b_k \setminus a_k) - v(S \cup B \setminus A)$. Define $\tilde{S} = S \cup b_k \setminus a_k$. If we can prove that the graph $\tilde{G}$ defined by left nodes $A \setminus a_k$, right nodes $B \setminus b_k$ and weights $\tilde{w}_{a_i b_j} = v(\tilde{S}) - v(\tilde{S} \cup b_j \setminus a_i)$ has $(a_1, b_1), \ldots, (a_{k-1}, b_{k-1})$ as the unique minimum weight matching and moreover $\tilde{w}_{a_i b_i} = w_{a_i b_i}$ then we can apply the induction hypothesis and conclude that:

$$v(S \cup b_k \setminus a_k) - v(S \cup B \setminus A) = \sum_{i=1}^{k-1} \tilde{w}_{a_i b_i} = \sum_{i=1}^{k-1} w_{a_i b_i}$$

In order to $(a_1, b_1), \ldots, (a_{k-1}, b_{k-1})$ is the unique minimum matching, first we bound $\tilde{w}_{a_i b_j}$ and then we show that any other matching has strictly larger weight.

$$\begin{aligned}
\tilde{w}_{a_i b_j} &= v(S \cup b_k \setminus a_k) + v(S) - [v(S \cup \{b_k, b_j\} \setminus \{a_k, a_i\}) + v(S)] \\
&\overset{*}{\geq} v(S \cup b_k \setminus a_k) + v(S) - \\
&\qquad \max\{v(S \cup b_j \setminus a_i) + v(S \cup b_k \setminus a_k), v(S \cup b_j \setminus a_k) + v(S \cup b_k \setminus a_i)\} \\
&= \min\{w_{a_i b_j}, w_{a_i b_k} + w_{a_k b_j} - w_{a_k b_k}\}
\end{aligned}$$

where the $(*)$ inequality follows from Lemma 4.3.

Now, given a matching $\tilde{M}$ different then $(a_1, b_1), \ldots, (a_{k-1}, b_{k-1})$ on $\tilde{G}$, construct an auxiliary graph in which we add the following edges for each $(a_i, b_j) \in \tilde{M}$: (i) if $\tilde{w}_{a_i b_j} = w_{a_i b_j}$, then we add an edge between a_i and b_j with weight $w_{a_i b_j}$ and sign $+1$. (ii) if $\tilde{w}_{a_i b_j} = w_{a_i b_k} + w_{a_k b_j} - w_{a_k b_k}$ we add edges between a_i and b_k with weight $w_{a_i b_k}$ and sign $+1$, an edge between a_k and b_j with weight $w_{a_k b_j}$ and sign $+1$ and one edge between a_k and b_k with weight $w_{a_k b_k}$ and sign -1. By a simply counting argument, the *signed* degree of each node a_i or b_i with $i < k$ is 1 and the signed degree of nodes a_k, b_k is 0. Now we argue that the total signed weight of this graph is at least $\sum_{i=1}^{k-1} w_{a_i b_i}$. Indeed, if there are no edges incident to k this is obvious since M was the *unique* minimum matching in G. If there are edges incident to k, consider a cycle C containing edge (a_k, b_k) in the union between the M (with weight $w_{a_i b_i}$) and the $+1$-signed edges in the auxiliary graph. Let C_M be the edges in the cycle belonging to M and let $C_{\tilde{M}}$ be the remaining edges. Note the the total weight of C_M is strictly smaller then the total weight of $C_{\tilde{M}}$ since M is the unique minimum matching. Therefore, we remove the edges in $C_{\tilde{M}}$ from the auxiliary graph and add the edges in C_M, where adding an edge (a_k, b_k) with $+1$ sign is equivalent in removing one edge (a_k, b_k) with -1 sign. By repeating this procedure we eventually obtain an auxiliary graph with strictly smaller weight then the original and no incident edges on a_k, b_k. The weight of such graph must be at least $\sum_{i=1}^{k-1} w_{a_i b_i}$ since M is the unique minimum matching.

In order to finish the proof, we just need to argue that $\tilde{w}_{a_i b_i} = w_{a_i b_i}$. By the previous argument: $\tilde{w}_{a_i b_i} \geq \min\{w_{a_i b_i}, w_{a_k b_i} + w_{a_i b_k} - w_{a_k b_k}\} = w_{a_i b_i}$ since by the minimality of matching M, $w_{a_i b_i} + w_{a_k b_k} < w_{a_k b_i} + w_{a_i b_k}$. For the other direction, we again use Lemma 4.3 to see that:

$$v(S \cup b_i \setminus a_i) + v(S \cup b_k \setminus a_k) \leq \max\{v(S) + v(S \cup \{b_i, b_k\} \setminus \{a_i, a_k\}),$$
$$v(S \cup b_k \setminus a_i) + v(S \cup b_i \setminus a_k)\} = v(S) + v(S \cup \{b_i, b_k\} \setminus \{a_i, a_k\})$$

since $w_{a_i b_i} + w_{a_k b_k} < w_{a_k b_i} + w_{a_i b_k}$ implies that $v(S \cup b_i \setminus a_i) + v(S \cup b_k \setminus a_k) > v(S \cup b_k \setminus a_i) + v(S \cup b_i \setminus a_k)$.

□

The next theorem provides an algorithm for obtaining a cycle of negative weight satisfying the Unique Minimum Weight Matching Condition starting from an arbitrary cycle of negative weight.

LEMMA 10.2. *Given a cycle C of negative weight in the exchange graph, then if for some i, $M^i = \{(a, b) \in C; a \in S_i'\}$ is not the Minimum Weight Matching between $A^i = \{a \in [n]; (a, b) \in M^i\}$ and $B^i = \{b \in [n]; (a, b) \in M^i\}$, then there is cycle C' of negative weight with strictly smaller number of edges.*

Proof. Assume that there is an alternative perfect matching

$$M' = \{(a^i_{j_1}, b^i_{j_2}), (a^i_{j_2}, b^i_{j_3}), \ldots, (a^i_{j_k}, b^i_{j_1})\}$$

with total weight no larger then the original one. Consider now k cycles in graph G where the t-th cycle C_t is formed by edge $(a^i_{j_t}, b^i_{j_{t+1}})$ and the path from $b^i_{j_{t+1}}$ to $a^i_{j_t}$ in the original cycle C. Each cycle C_i is either C (if this edge of M' is also in M^i) or is a cycle with smaller number of edges (otherwise).

Next we present a counting argument, there exists an integer ℓ such that the multiset union of cycles $\{C_i; C_i \neq C\}$ has ℓ copies of each edge in $C \setminus M^i$, $\ell - 1$ copies

of each edge in M^i and one copy of each edge in M'. The argument is as follows: transverse the cycle in the following way: start in $b^i_{j_1}$ and follow cycle C until $a^i_{j_k}$, then take an edge $(a^i_{j_k}, b^i_{j_k})$ in M^i and transverse the cycle from $b^i_{j_k}$ to $a^i_{j_{k-1}}$, take then the edge $(a^i_{j_{k-1}}, b^i_{j_{k-1}})$ in M^i and then transverse C from $b^i_{j_{k-1}}$ to $a^i_{j_{k-2}}$, continuing the same procedure until we take edge (a^i_1, b^i_1) in which case we complete an integral number ℓ of loops around the cycle. The transversal of from $b^i_{j^{t+1}}$ to $a^i_{j^t}$ corresponds to the edges of cycle C_t minus the edge in M'. So if we look at the edges transversed add M' and subtract M^i, we get exactly the multi-set of edges in the union of the cycles C_t.

Therefore, the sum of weights of such cycles is at most ℓ times the sum of weights in C and therefore negative. Then there must exist some cycle $C_i \neq C$ of negative weight, contradicting that C has minimal number of edges among all negative cycles. □

ALGORITHM 4: Cycle canceling

Input: gross substitute valuations $v^1, \ldots, v^m : 2^{[n]} \to \mathbb{R}_+$
Initialize with an arbitrary partition $S_1, \ldots, S_m$ of $[n]$
Define G implicitly as the exchange graph corresponding to the current allocation
while G has negative weight cycles
  find a negative cycle C satisfying the Unique Min Matching Cond.
  let $A^i = \{a; (a,b) \in C, a \in S_i\}$ and $B^i = \{b; (a,b) \in C, a \in S_i\}$
  update $S_i = S_i \cup B^i \setminus A^i$ for all i.

The previous discussion suggests an algorithm that strictly improves an allocation, yet it doesn't still guarantee polynomial running time. In what follows we show that a careful choice of cycles and will be enough to guarantee polynomial runtime. This will be done by measuring the progress in terms of a suitable potential function.

10.3.2. Polynomial running time via canceling minimum mean weight cycles. The cycles we will consider are the cycles with minimum mean weight, i.e., cycles minimizing: $w(C)/|C|$, where $w(C) = \sum_{e \in C} w_e$. A minimum mean cycle in a directed graph can be found in time $O(N \cdot E)$ where N is the number of nodes and E is the number of edges using an algorithm[5] due to Karp [15]. We note in particular that the algorithm in Lemma 10.2 can be used to find a cycle minimum mean weight cycle C_t satisfying the Unique Minimum Matching Condition, since the cycles C_t produced by the algorithm suggested by the lemma are such that:

$$\min_t \frac{w(C_t)}{|C_t|} \leq \frac{\sum_t w(C_t)}{\sum_t |C_t|} = \frac{\ell \cdot w(C) - w(M^i) + w(M')}{\ell \cdot |C|} \leq \frac{\ell \cdot w(C)}{\ell \cdot |C|} \qquad \text{(CD)}$$

This completes the description of the Minimum mean weight cycle canceling algorithm. In order to show that it runs in weakly polynomial time, we bound the number of iterations using a potential function.

Given an allocation $S_1, \ldots, S_m$ we define the slackness $\epsilon(S_1, \ldots, S_m)$ as the minimum $\epsilon \geq 0$ such that there exist a price p, such that for all agents i, $a \in S'_i$ and $b \notin S'_i$

[5]Sketch of Karp's algorithm for the minimum mean cycle problem: given a strongly-connected weighted directed graph, fix an arbitrary source s and compute via dynamic programming the minimum weight path of length k from s to v for all v, using the recursion $F_k(v) = \min_u F_k(u) + w_{uv}$. Then the weight of the minimum weight cycle is given by $\mu = \min_v \max_{0 \leq k < n} (F_n(v) - F_k(v))/(n-k)$.

(possibly dummy items),

$$v^i(S_i' \cup b \setminus a) - p_b + p_a \leq v(S_i') + \epsilon.$$

A solution is optimal iff $\epsilon(S_1, \ldots, S_m) = 0$. In some sense, this function measures how sub-optimal an allocation is. The presentation is an adaptation of the proof of Goldberg and Tarjan [10] for the minimum cost circulation algorithm in strongly polynomial time via cycle canceling.

The first lemma relates ϵ to the value of the minimum mean-weight cycle:

LEMMA 10.3 (Goldberg, Tarjan [10]). *Given a certain allocation $S_1, \ldots, S_m$, then $\epsilon(S_1, \ldots, S_m) = -\mu$ where μ is the value of the minimum mean weight cycle in the exchange graph corresponding to this allocation.*

Proof. This can be obtained by writing the natural Linear Program that computes ϵ and taking the dual. Given a directed graph (N, E) with costs c_{ij} on edges, consider the program:

$$\min \epsilon \quad \text{s.t.} \quad c_{ij} - p_i + p_j \leq \epsilon, \forall (i,j) \in E$$

By the duality theorem, this corresponds to the solution of the following Linear Program:

$$\max \sum_{(i,j)\in E} -c_{ij} x_{ij} \quad \text{s.t.} \quad \begin{array}{l} \sum_{k;(i,k)\in E} x_{ik} = \sum_{k;(k,i)\in E} x_{ki}, \forall i \in N \\ \sum_{(i,j)\in E} x_{ij} = 1 \\ x_{ij} \geq 0, \forall (i,j) \in E \end{array}$$

which corresponds to the circulation of minimum mean-weight. Since each circulation $x \in \mathbb{R}_+^E$ can be decomposed in weighted sum of cycles $x = \sum_t \alpha_t \cdot z^t$ where $\alpha_t \in \mathbb{R}_+$ and $z^t \in \mathbb{R}_+^E$ is the indicator vector of a cycle. Therefore, there must be a mean-weight circulation that is a cycle and this must be the mean-weight cycle. □

The following corollary follows from the previous proof and complementarity slackness:

COROLLARY 10.4. *Given a certain allocation $S_1, \ldots, S_m$ and a price vector with respect to which the allocation is $\epsilon(S_1, \ldots, S_m)$-optimal, then if C is the mean weight minimum cycle, then along the edges (a, b) of the cycle, $w_{ab} + p_a - p_b = -\epsilon$. In particular, if the cycle satisfies the Unique Minimum Weight Matching Condition, performing the changes prescribed by the cycle produces an improvement of $\epsilon \cdot |C|$ to the value of the allocation.*

The next lemma uses this fact to shows that canceling a mean weight negative cycle in Algorithm 4 doesn't increase the slackness $\epsilon(S_1, \ldots, S_m)$.

LEMMA 10.5. *Given an allocation $S_1, \ldots, S_m$, then if C is a minimum mean weight cycle satisfying the Unique Minimum Weight Condition, then performing the exchanges prescribed by the cycle can't reduce $\epsilon(S_1, \ldots, S_m)$.*

Proof. Let $\epsilon = \epsilon(S_1, \ldots, S_m)$ and p be a price vector with respect to which the allocation is optimal. Let S'_i be the current allocation (augmented with the i-th dummy item) and $S''_i = S_i \cup B^i \setminus A^i$ using the notation of Algorithm 4.

By ϵ-optimality, we know that $v^i(S'_i) - v^i(S_i \cup b \setminus a) - p_a + p_b \geq -\epsilon$ for all $a \in S'_i$ and $b \notin S'_i$. Fix some i and define price vector p' such that $p'_a = p_a$ for $a \in S_i$ and $p'_b = p_b + \epsilon$ for $b \notin S_i$. This gives us $v^i(S'_i) - v^i(S_i \cup b \setminus a) - p'_a + p'_b \geq 0$. Hence by Theorem 3.4, $v^i(S'_i) - p(S'_i) = v^i(S'_i) - p'(S'_i) \geq v^i(T) - p'(T)$ for all $T \subseteq [n+m]$. This gives us

$$\begin{aligned} v^i(S''_i) - p(S''_i) = v^i(S'_i) - p(S'_i) + \epsilon \cdot |A^i| &\geq v^i(S''_i \cup b \setminus a) - p'(S''_i \cup b \setminus a) + \epsilon \cdot |A^i| \\ &\geq v^i(S''_i \cup b \setminus a) - p(S''_i \cup b \setminus a) - \epsilon \end{aligned}$$

where the equality comes from the complementarity conditions in the previous lemma, the first inequality comes with the previous inequality with $T = S''_i \cup b \setminus a$ and the second inequality comes from the definition of p'. The inequality obtained says that for the new allocation, the same price p proves that the slackenss $\epsilon(S''_1, \ldots, S''_m)$ is at most ϵ. □

The previous two lemmas already give us a weakly polynomial time bound on the running time of Algorithm 4. Let $S^*_1, \ldots, S^*_m$ be the optimal allocation, $S_1, \ldots, S_m$ be an arbitrary allocation. Then we know by the definition of ϵ optimality and by Theorem 3.4 that:

$$\textsc{Gap}(S_1, \ldots, S_m) := \sum_i v^i(S^*_i) - \sum_i v^i(S_i) \leq n \cdot \epsilon(S_1, \ldots, S_m)$$

Since after one iteration, the allocation improves by at least $\epsilon \cdot |C|$, $\textsc{Gap}(S_1, \ldots, S_m)$ goes down by at least a factor of $1 - 1/n$. Since the initial gap is at most $M = \sum_i v^i([n])$, then in $n \log(nM)$ iterations, the gap is at most $1/n$, which means that the solution is optimal. Each iteration consists in building the exchange graph, which takes time $O(T \cdot (n+m)^2)$, where T is the cost of evaluating the valuation function on any given set; and finding the minimum mean weight cycle, which takes time $O((n+m)^3)$. This gives an overall running time of $O((T \cdot (n+m)^2 + (n+m)^3) \cdot n \cdot \log(n \cdot M))$.

10.3.3. Cycle canceling in strongly polynomial time. Finally, we use a variation of the minimum mean weight cycle canceling algorithm due to Zimmermann [33] to make the running time to strongly polynomial time. Consider a vector β with components $\beta_{ij} \in \{0, 1\}$ for $i \in [n]$ and $j \in [n+m]$ where i represents an agent and j represents a (possibly dummy) item. We say that an allocation $S'_1, \ldots, S'_m$ is (ϵ, β)-optimal iff there is a price vector p such that for all $a \in S'_i$ and $b \notin S'_i$ it holds that:

$$v^i(S'_i \cup b \setminus a) - p_b + p_a \leq v(S'_i) + \epsilon \cdot \beta_{ib}$$

We define the β-slackness $\epsilon_\beta(S_1, \ldots, S_m)$ as the smallest ϵ such that an allocation is (ϵ, β)-optimal.

Similarly to Lemma Lemma 10.3, the value of $\epsilon_\beta(S_1, \ldots, S_m)$ for a given vector β is equivalent to a negative cycle problem. The next lemma can be obtained following the proof of Lemma 10.3 substituting ϵ by $\epsilon\beta_{ij}$ in the primal program and taking the corresponding dual.

LEMMA 10.6. *Consider an allocation $S_1, \ldots, S_m$ and a binary vector β such that every negative cycle in the exchange graph has at least one edge (a, b), $a \in S'_i$, $b \notin S'_i$ with $\beta_{ib} = 1$. Then $\epsilon(S_1, \ldots, S_m) = -\mu_\beta$ where $\mu_\beta = \min_{cycle\ C} w(C)/\beta(C)$ where $\beta(C) = \sum_{(a,b)\in C} \beta_{i(a),b}$ where $i(a)$ is the agent holding item a in the current allocation.*

One can compute such cycle using Megiddo's technique [22] for solving combinatorial optimization problems with rational objective function. As a brief overview, Megiddo's technique consists in solving the problem of deciding if a graph has a negative weight cycle (which can be solved, for example, using the Bellman-Ford algorithm) using weights parametrized by a parameter t, i.e., $w_e(t) = w_e(t) - t \cdot \beta_e(t)$. This can be solved in strongly polynomial time by running the standard Bellman-Ford algorithm using the following observation: the algorithm just performs additions and multiplications by scalar and comparisons. The first two operations can be done on the parametrized weights. For comparisons, one can augment the Bellman-Ford algorithm with an interval I of parameters initialized by $(-\infty, \infty)$. Once a comparison is performed, the algorithm tries to decide if the comparison evaluation in the same way for all parameters in the interval. If so, the comparison is decided. If not, the interval is split and the computation progresses independently in each interval. This method gives the solution for all parameters. Then, the cycle with smallest $w(C)/\beta(C)$ corresponds to the largest parameter t for which there is no negative cycle.

By the same argument as before, it is possible to find a cycle minimizing the ratio $w(C)/\beta(C)$ while satisfying the Unique Minimum Weight Condition. It follows from the same argument used in Lemma 10.1 – essentially by substituting $|C_t|$ by $\beta(C_t)$ in inequality (CD).

The definition of (ϵ, β)-optimality allows for a stronger version of Lemma 10.5 which measures progress by showing that if an allocation is (ϵ, β)-optimal, then after the algorithm improves the allocation by exchanging items along a minimum ratio cycle, then the allocation is (ϵ, β')-optimal for some stricted vector β', i.e., a vector such that $\beta' \leq \beta$ and $\beta' \neq \beta$.

LEMMA 10.7. *Given an (ϵ, β)-optimal allocation $S_1, \ldots, S_m$, then if C is a negative cycle minimizing $w(C)/\beta(C)$ satisfying the Unique Minimum Weight Condition, then performing the exchanges prescribed by the cycle will result in an (ϵ, β')-allocation for where for $j \in S_i$, $\beta'_{ij} = 0$ and for $j \notin S_i$, $\beta'_{ij} = \beta_{ij}$.*

Proof. The proof follows the same structure and notation of Lemma 10.5. Let p be a price vector for which the original allocation is (ϵ, β)-optimal. Therefore: $v^i(S'_i) - v^i(S'_i \cup b \setminus a) - p_a + p_b \geq -\epsilon \cdot \beta_{ib}$ for all $a \in S'_i$ and $b \notin S'_i$. Define price vector p' such that $p'_j = p_j + \beta_{ij}$ for $j \notin S'_i$ and $p'_j = p_j$ otherwise. Then following the arguments in Lemma 10.5 and denoting $\beta(A^i) = \sum_{j\in A^i} \beta_{ij}$, we get:

$$\begin{aligned} v^i(S''_i) - p(S''_i) &= v^i(S'_i) - p(S'_i) + \epsilon \cdot \beta(A^i) \\ &\geq v^i(S''_i \cup b \setminus a) - p'(S''_i \cup b \setminus a) + \epsilon \cdot \beta_i(A^i) \\ &\geq v^i(S''_i \cup b \setminus a) - p(S''_i \cup b \setminus a) - \epsilon \cdot \beta^i((S''_i \cup b \setminus a) \setminus S'_i) + \epsilon \cdot \beta_i(A^i) \end{aligned}$$

If $b \notin S'_i$, then: $\beta^i((S''_i \cup b \setminus a) \setminus S'_i) - \beta_i(A^i) \leq \beta_{ib}$. If $b \in S'_i$ then: $\beta^i((S''_i \cup b \setminus a) \setminus S'_i) - \beta_i(A^i) \leq 0$. □

COROLLARY 10.8. *In the setting of the previous lemma, after the exchanges are performed, either the new allocation is optimal, or in all negative cycles of the new exchange graph there is at least one edge* (a, b) *with* $a \in S'_i$ *and* $\beta'_{ib} = 1$.

Proof. If C is a negative cycle in the resulting graph then summing the inequalities $w_{ab} + p_b - p_a \geq -\epsilon\beta'_{i(a)b}$, then we get $0 > w(C) \geq -\epsilon\beta'(C)$, so $\beta'(C) > 0$. □

The previous lemmas suggest the following algorithm: intialize β with $\beta_{ib} = 1$ for all $i \in [m]$ and $b \in [n + m]$. Then run the cycle canceling algorithm picking the cycle minimizing $w(C)/\beta(C)$. After canceling the cycle, update β by setting to zero the components β_{ib} according to Lemma 10.7.

Corollary 10.8 guarantees that while the allocation is sub-optimal there will be a negative cycle with positive β-value, so the algorithm finishes on an optimal allocation. Given that the chosen cycle in each iteration has at least one edge corresponding to a 1-enty and this is set to zero in the end of each iteration, there are at most $(n + m)^2$ iterations. Computing a minimum ratio cycle can be done in time $O(E \cdot N^2 \log(N))$ where N is the number of edges and E is the number of edges. If T is the cost of evaluating a valuation function, the overall complexity of the algorithm is: $O((T \cdot (n + m)^4 + (n + m)^6 \log(n + m)))$.

THEOREM 10.9 (Murota [25]). *The algorithm that finds a minimum weight mean cycle with minimum number of edges among such cycles (Algorithm 4) is a strongly polynomial time algorithm for the gross substitute welfare problem.*

We remark that Murota [25] gives a strongly polynomial time algorithm for a more general problem called the *Valuated Matroid Intersection* problem. In this problem there is a bipartite graph defined by (U, V, E) a weight function $w : E \to \mathbb{R}$ and gross substitute functions $v^U : 2^U \to \mathbb{R}$ and $v^V : 2^V \to \mathbb{R}$, find the matching M maximizing $w(M) + v^U(M^U) + v^V(M^V)$ where M^U and M^V are the subsets of U and V covered by the matching. The techniques for solving this problem are very similar to the ones presented in this survey: they involve building an exchange graph and improving the allocation along suitable cycles. Besides the cycle canceling approach, Murota also proposes an additional strongly-polinomial time algorithm based on a flow-like primal-dual approach.

Acknowledgments. The author thanks Noam Nisan and Shahar Dobzinski for the initial discussions that led to this survey. He is also in debt with Moshe Babaioff and Oren Ben-Zwi for many detailed comments and suggestions to improve this manuscript. Thanks to Oren for providing a fix to the proof of Theorem 3.4 in an earlier version of this manuscript. The author also thanks Michael Ostrovsky, Brendan Lucier and Tim Roughgarden for suggestions and conversations in which I clarified some of the points in this survey.

REFERENCES

[1] L. Ausubel and P. Milgrom. Ascending auctions with package bidding. *Frontiers of Theoretical Economics*, 1(1), 2002.

[2] D. P. Bertsekas. The auction algorithm: a distributed relaxation method for the assignment problem. *Ann. Oper. Res.*, 14(1-4):105–123, June 1988.

[3] S. Bikhchandani and J. W. Mamer. Competitive equilibrium in an exchange economy with indivisibilities. *Journal of Economic Theory*, 74(2):385–413, June 1997.

[4] M. Bing, D. J. Lehmann, and P. Milgrom. Presentation and structure of substitutes valuations. In *ACM Conference on Electronic Commerce*, pages 238–239, 2004.

[5] G. Demange, D. Gale, and M. Sotomayor. Multi-item auctions. *Journal of Political Economy*, 94(4):863–72, August 1986.

[6] A. Dress and W. Terhalle. Rewarding maps: On greedy optimization of set functions. *Advances in Applied Mathematics*, 16(4):464 – 483, 1995.

[7] A. Dress and W. Terhalle. Well-layered mapsa class of greedily optimizable set functions. *Applied Mathematics Letters*, 8(5):77 – 80, 1995.

[8] A. W. Dress and W. Wenzel. Valuated matroids: a new look at the greedy algorithm. *Applied Mathematics Letters*, 3(2):33 – 35, 1990.

[9] S. Fujishige and Z. Yang. A note on kelso and crawford's gross substitutes condition. *Math. Oper. Res.*, 28(3):463–469, July 2003.

[10] A. V. Goldberg and R. E. Tarjan. Finding minimum-cost circulations by canceling negative cycles. *Journal of the ACM (JACM)*, 36(4):873–886, 1989.

[11] F. Gul and E. Stacchetti. Walrasian equilibrium with gross substitutes. *Journal of Economic Theory*, 87(1):95–124, July 1999.

[12] F. Gul and E. Stacchetti. The english auction with differentiated commodities. *Journal of Economic Theory*, 92(1):66–95, May 2000.

[13] J. W. Hatfield and P. R. Milgrom. Matching with Contracts. *American Economic Review*, 95(4):913–935, September 2005.

[14] J. E. Hopcroft and R. M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM J. Comput.*, 2(4):225–231, 1973.

[15] R. M. Karp. A characterization of the minimum cycle mean in a digraph. *Discrete mathematics*, 23(3):309–311, 1978.

[16] J. Kelso, Alexander S and V. P. Crawford. Job matching, coalition formation, and gross substitutes. *Econometrica*, 50(6):1483–1504, November 1982.

[17] M. Klein. A primal method for minimal cost flows with applications to the assignment and transportation problems. *Management Science*, 14(3):205–220, 1967.

[18] B. Korte, L. Lovász, and R. Schrader. *Greedoids*. Algorithms and Combinatorics. Springer-Verlag, 1991.

[19] H. W. Kuhn. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly*, 2(1–2):83–97, March 1955.

[20] E. Lawler. *Combinatorial optimization: networks and matroids*. Dover Books on Mathematics Series. Dover Publications, Incorporated, 1976.

[21] B. Lehmann, D. J. Lehmann, and N. Nisan. Combinatorial auctions with decreasing marginal utilities. *Games and Economic Behavior*, 55(2):270–296, 2006.

[22] N. Megiddo. Combinatorial optimization with rational objective functions. *Mathematics of Operations Research*, 4(4):414–424, 1979.

[23] K. Murota. Convexity and steinitz's exchange property. *Advances in Mathematics*, 124(2):272 – 311, 1996.

[24] K. Murota. Valuated matroid intersection i: Optimality criteria. *SIAM J. Discrete Math.*, 9(4):545–561, 1996.

[25] K. Murota. Valuated matroid intersection ii: Algorithms. *SIAM J. Discrete Math.*, 9(4):562–576, 1996.

[26] K. Murota and A. Shioura. M-convex function on generalized polymatroid. *Mathematics of Operations Research*, 24(1):pp. 95–105, 1999.

[27] N. Nisan and I. Segal. The communication requirements of efficient allocations and supporting prices. *J. Economic Theory*, 129(1):192–224, 2006.

[28] J. Oxley. *Matroid theory*. Oxford Graduate Texts in Mathematics Series. Oxford University Press, Incorporated, 1992.

[29] H. Reijnierse, A. v. Gellekom, and J. A. M. Potters. Verifying gross substitutability. *Economic Theory*, 20(4):pp. 767–776, 2002.

[30] A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Number v. 1 in Algorithms and Combinatorics. Springer, 2003.

[31] L. Shapley. *Complements and Substitutes in the Optimal Assignment Problem*. ASTIA document. Rand Corporation, 1958.

[32] L. Walras. *Elements of Pure Economics: Or the Theory of Social Wealth*. Elements of Pure Economics, Or the Theory of Social Wealth. Taylor & Francis, 2003.

[33] U. Zimmermann. Negative circuits for flows and submodular flows. *Discrete applied mathematics*, 36(2):179–189, 1992.