Notes on Border's Theorem

1 Background

Consider a setting where an object needs to be allocated to n agents. Each agent i has a private value v_i for the object. Assume that the value of each agent i is drawn using some distribution F_i with some support V_i. A planner/designer sets up an *allocation rule* to allocate the object. Such an allocation rule assigns probabilities of winning the object to each agent at every profile of values $(v_1, \ldots, v_n)$. Given such an allocation rule, an agent i with value v_i can compute her *interim* allocation probability of winning the object. This induces an interim allocation rule, which maps each $v_i \in V_i$ to a number in $[0, 1]$. Thus, every allocation rule induces n interim allocation rules.

These interim allocation rules form the heart of many problems in economic theory. For instance, if the allocation rule is used in an auction, the incentive constraints can be reformulated to be just monotonicity constraints of interim allocation rules and the objective function of a revenue-maximizing seller can be written in terms of interim allocation rules.

Suppose we are given n arbitrary maps: from V_i to $[0, 1]$ for each i. Border's theorem asks if such maps can correspond to n valid interim allocation rules. That is, if there exists an allocation rule which can generate these n maps as interim allocation rules. The initial motivation for such a question came from auction theory: early necessary conditions and conjectures were due to Matthews and Maskin. However, Border's theorem now has applicability in other areas of economic theory.

There are two (related) versions of Border's theorem: (1) with a finite type space; (2) with a continuum type space. This note covers both the versions. We start with a symmetric model and finite type space. We then extend this finite type space characterization to a model without symmetry. We then look a continuum type space with symmetry and without symmetry. We then give some applications of Border's theorem.

2 The symmetric model

A single object is to be allocated to a set of n agents. The set of agents are denoted by N. Agents are ex-ante identical: each agent i draws its values from some finite set X and

the corresponding probability distribution is F. So, $F(x) \in [0,1]$ is the probability that an agent has value $x \in X$, and $\sum_{x \in X} F(x) = 1$. A profile of values is n draws from X using the same distribution F.

It may be useful to represent a profile of values alternatively. A **counting function** is a map $c : X \to \{0, 1, \ldots, n\}$ such that

$$\sum_{x \in X} c(x) = n$$

Let $\mathcal{C}$ be the set of all counting functions. Each counting function $c \in \mathcal{C}$ represents a profile of values (and all its permutations): if a value x belongs to this profile if and only if $c(x) > 0$. Further, $c(x)$ represents the number of agents who have value x. A counting function is thus an *anonymous* representation of a profile of values since it only counts the number of occurrences of a particular value.

We are now ready to define a symmetric allocation rule.

DEFINITION **1** *A* **symmetric allocation rule** *is a map* $q : X \times \mathcal{C} \to [0,1]$ *such that*

$$\begin{aligned} q(x,c) &= 0 \qquad && \text{if } c(x) = 0 \\ \sum_{x \in X} c(x) q(x,c) &\leq 1 \qquad && \forall\ c \in \mathcal{C} \end{aligned}$$

A symmetric allocation rule assigns all agents with identical value the object with identical probability. Hence, $q(x,c)$ defines the probability with which an agent of type x receives the object when any valuation profile corresponding to count function c occurs. The first constraint ensures that the object is only given to values that appear in the profile. The second condition ensures that the total probability is less than 1.

Let $\lambda(c)$ be the probability with which a profile of values corresponding to counting function c is realized: this is the sum of probabilities of all profile of values that generate the counting function c. This probability can be computed from F as:

$$\lambda(c) = \prod_{x \in X} [F(x)]^{c(x)} \qquad \forall\ c \in \mathcal{C}$$

We are now ready to introduce interim allocation rules.

DEFINITION **2** *An* **interim allocation rule** *is a map* $Q : X \to [0, 1]$.

The interpretation of $Q(x)$ is the (interim) probability with which an agent with value x wins the object (in some allocation rule). Of course, every allocation rule q generates an interim allocation rule Q as follows.

$$nQ(x)F(x) = \sum_{c \in \mathcal{C}} c(x)q(x, c)\lambda(c) \qquad \forall\ x \in X \tag{1}$$

The LHS of equation (1) calculates (using Q) the probability that winner is an agent with value x. The RHS calculates the same thing using q. It may be useful to define $\widetilde{Q}$ from Q as follows:

$$\widetilde{Q}(x) = nQ(x)F(x)$$

Hence, (1) can be rewritten as follows:

$$\widetilde{Q}(x) = \sum_{c \in \mathcal{C}} c(x)q(x, c)\lambda(c) \qquad \forall\ x \in X \tag{2}$$

DEFINITION **3** *An interim allocation rule* $Q : X \to [0, 1]$ *is* **implementable** *if there exists an allocation rule* q *such that equation (2) (or, equivalently (1)) holds.*

This leads to our first theorem.

THEOREM **1 (Finite Symmetric Border)** *An interim allocation rule* Q *is implementable if and only if*

$$\sum_{x \in S} \widetilde{Q}(x) \leq 1 - \Big(\sum_{x \in X \setminus S} F(x) \Big)^n \qquad \forall\ S \subseteq X \tag{3}$$

Proof: We first construct a layered network as follows. There is a source s and a sink vertex t. First layer consists of all the types in X, i.e., a vertex for each type $x \in X$. Second layer consists of all the counting functions $\mathcal{C}$, i.e., a vertex for each counting function $c \in \mathcal{C}$.

There is an edge (s, x) for each $x \in X$ and there is an edge (c, t) for each $c \in \mathcal{C}$. There is an edge (x, c) if and only if $c(x) > 0$. Capacity of edges (x, c) is infinite: $\kappa(x, c) := \infty$. Capacity

of edge (s, x) is $\kappa(s, x) := \widetilde{Q}(x) = nQ(x)F(x)$. Capacity of edge (c, t) is $\kappa(c, t) := \lambda(c)$. A representative network is shown in Figure 1.

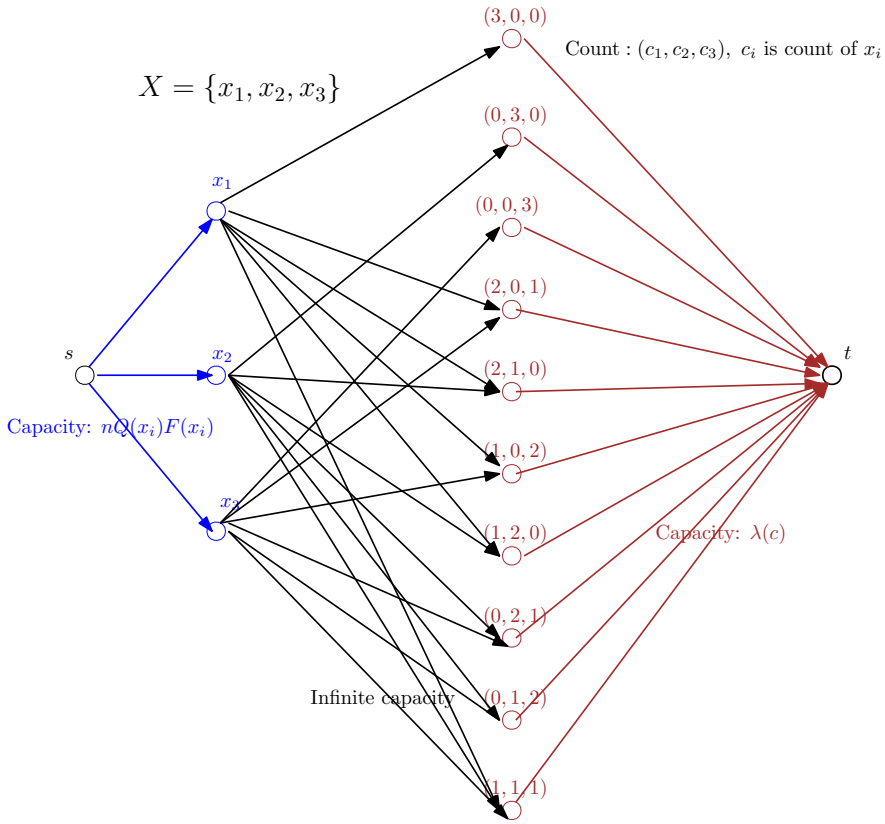

Figure 1: Network corresponding to three types

Flow defines $f(e)$ for every edge e such that capacity constraints and flow balancing constraints hold. The first step of the proof is to show that Q is implementable if and only if there is a "maximal" flow f such that $f(s, x) = \widetilde{Q}(x) = nQ(x)F(x)$ for all x.

Necessity. Suppose Q is implementable by q. Then, define the following flow:

$$\begin{aligned} f(s,x) &= \widetilde{Q}(x) = \kappa(s,x) \qquad \forall\ x \in X \\ f(x,c) &= c(x)q(x,c)\lambda(c) \leq \kappa(x,c) \qquad \forall\ x,c \\ f(c,t) &= \sum_{x:c(x)>0} c(x)q(x,c)\lambda(c) \leq \lambda(c) = \kappa(c,t) \qquad \forall\ c \end{aligned}$$

So, by definition, f satisfies capacity constraints. For flow balancing, we see that for any vertex x in the first layer, the incoming flow is $\widetilde{Q}(x)$. The outgoing flow is

$$\sum_{c:c(x)>0} c(x)q(x,c)\lambda(c) = \widetilde{Q}(x)$$

Hence, flow balancing holds at x. Now, pick c. Outgoing flow is

$$\sum_{x:c(x)>0} c(x)q(x,c)\lambda(c)$$

which is also the sum of incoming flows. So, this is a feasible flow. Since the edges (s,x) for each x flow at full capacity, it is a maximal flow.

Sufficiency. Suppose there is a maximal flow f where each edge (s,x) flows at full capacity. Then, define q from f as follows. For every (x,c) with $c(x) > 0$,

$$q(x,c) := \frac{f(x,c)}{c(x)\lambda(c)}$$

If $c(x) = 0$, we set $q(x,c) = 0$. We show that q is feasible. First, for every x, flow balancing at x gives

$$nQ(x)F(x) = f(s,x) = \sum_{c:c(x)>0} f(x,c) = \sum_{c:c(x)>0} c(x)q(x,c)\lambda(c)$$

Hence, q generates the interim probabilities Q. By definition $q(x,c) \geq 0$. By flow balancing

at c, we see that

$$\sum_{x:c(x)>0} f(x,c) = \sum_{x:c(x)>0} c(x)q(x,c)\lambda(c) = f(c,t) \leq \lambda(c)$$

where the last inequality is because of capacity constraint on (c,t). As a result, we have for every c,

$$\sum_{x:c(x)>0} c(x)q(x,c) \leq 1$$

which is the desired feasibility constraint. This completes the first step of the proof.

The next step shows that Border's (symmetric) inequalities hold if and only if there is a maximal flow f where each edge (s,x) flows at full capacity. This in turn is true if and only if the "the cut" where source s is on one side and all other vertices are on the other side is a "minimum cut", where we explain the meanings of cut and minimum cut below.

For this, we use the max-flow equals min-cut theorem. First, a cut is a partitioning of set of vertices such that s and t are on different sets. Second, the capacity of a cut is the capacity of all the edges that go from s side of the cut to t side of the cut. Given this definition, the candidate minimum cuts must have no vertex x on s-side and a corresponding c on t side such that $c(x) > 0$. This is because the capacity of these edges is infinite. Hence, we only consider cuts where some $S \subseteq X$ set of types are on s-side and every c with $c(x) > 0$ for some $x \in S$ is also on s-side. The capacity of this cut is thus the capacities of all edges from the c vertices on s-side to sink t and all edges from s to vertices in $X \setminus S$:

$$n \sum_{x \in X \setminus S} Q(x)F(x) + \sum_{c:c(x)>0 \text{ for some } x \in S} \lambda(c)$$

Using the fact that $\sum_c \lambda(c) = 1$, we get

$$\sum_{c:c(x)>0 \text{ for some } x \in S} \lambda(c) = 1 - \sum_{c:c(x)=0 \ \forall \ x \in S} \lambda(c)$$

Hence, the total capacity is

$$n \sum_{x \in X \setminus S} Q(x)F(x) + 1 - \sum_{c: c(x)=0 \ \forall \ x \in S} \lambda(c)$$

The capacity of the cut involving $S = \emptyset$ is

$$n \sum_{x \in X} Q(x)F(x)$$

Hence, a necessary and sufficient condition for the $S = \emptyset$ cut to be minimal is

$$\begin{aligned} n \sum_{x \in X} Q(x)F(x) \leq n \sum_{x \in X \setminus S} Q(x)F(x) + 1 - \sum_{c: c(x)=0 \ \forall \ x \in S} \lambda(c) \\ \Leftrightarrow n \sum_{x \in S} Q(x)F(x) \leq 1 - \sum_{c: c(x)=0 \ \forall \ x \in S} \lambda(c) \end{aligned}$$

Finally, the probability of the event $c : c(x) = 0 \ \forall \ x \in S$ is the probability that type of each agent lies in $X \setminus S$. This probability is

$$\left[1 - \sum_{x \in S} F(x)\right]^n$$

Hence, the relevant necessary and sufficient condition is

$$n \sum_{x \in S} Q(x)F(x) \leq 1 - \left[1 - \sum_{x \in S} F(x)\right]^n$$

■

2.1 Submodularity and polymatroids

In continuous optimization problems, concave maximization plays an important role. A counterpart of concave maximization in the framework of discrete optimization is submodular optimization.

We study a class of optimization problems for which a *greedy* solution exists. Further, whenever the primitives of the problems are integral, then the greedy solution is also an inte-

gral solution. This illustrates an important class of problems where the *total unimodularity* property, a widely applied sufficient condition for existence of integral extreme points, need not hold, but still we get integral extreme points.

We will be talking about set functions. Let X be a finite set and $\mathcal{P}(X)$ be the set of all subsets of X. We will be interested in functions $\psi : \mathcal{P}(X) \to \mathbb{R}$, where we will normalize $\psi(\emptyset) = 0$ throughout.

To think of a real like example, think of firm producing a single product. To produce the first unit, it requires some investment in machines and other fixed costs. To produce the second unit, the cost is usually lower. As the production becomes large, there is natural economies of scale, and the extra unit of cost of production goes down. This idea is captured in the following definition.

DEFINITION **4** *A function $\psi : \mathcal{P}(X) \to \mathbb{R}$ is* **submodular** *if for all $A, B \subseteq X$, we have*

$$\psi(A \cup B) + \psi(A \cap B) \leq \psi(A) + \psi(B)$$

A function $\psi : \mathcal{P}(X) \to \mathbb{R}$ is **non-decreasing** *if for all $A, B \subseteq X$ with $A \subseteq B$, we have $\psi(A) \leq \psi(B)$.*

If a function $\psi : \mathcal{P}(X) \to \mathbb{R}$ is both submodular and non-decreasing it is called a **polymatroid**.

Consider an example with $X = \{a, b, c\}$ and $\psi(\{a\}) = \psi(\{b\}) = \psi(\{c\}) = 1$, $\psi(\{a, b\}) = 3, \psi(\{b, c\}) = 2, \psi(\{c, a\}) = 4$, $\psi(\{a, b, c\}) = 4$. The function ψ is non-decreasing. However, it is not submodular since $\psi(\{a\}) + \psi(\{b\}) = 2 < 3 = \psi(\{a, b\})$.

An alternate definition of a submodular function is the following.

THEOREM **2** *A set function $\psi : \mathcal{P}(X) \to \mathbb{R}$ is submodular if and only if for all $A, B \subseteq X$ with $A \subseteq B$ and for all $b \notin B$, we have*

$$\psi(B \cup \{b\}) - \psi(B) \leq \psi(A \cup \{b\}) - \psi(A)$$

The theorem says that the "marginal" contribution to a smaller subset is larger than to a bigger subset. This is consistent with the idea that the derivative of a concave function is non-increasing.

Proof: Suppose $\psi : \mathcal{P}(X) \to \mathbb{R}$ is a submodular function. Pick $A \subseteq B \subseteq X$ and $b \notin B$. Note that $(A \cup \{b\}) \cup B = B \cup \{b\}$ and $(A \cup \{b\}) \cap B = A$. Using submodularity we get, $\psi(A \cup \{b\}) + \psi(B) \geq \psi(B \cup \{b\}) + \psi(A)$, which gives the desired inequality.

For the converse, pick any $A, B \subseteq N$. If $B \subseteq A$, then $A \cup B = A$ and $A \cap B = B$. As a result, trivially, $\psi(A) + \psi(B) = \psi(A \cup B) + \psi(A \cap B)$. So, assume $B \nsubseteq A$, and let $B \setminus A := \{b_1, \ldots, b_k\}$. Now,

$$\begin{aligned}
\psi(A \cup B) - \psi(B) &= \psi(B \cup \{b_1, \ldots, b_k\}) - \psi(B) \\
&= \psi(B \cup \{b_1, \ldots, b_k\}) - \psi(B \cup \{b_1, \ldots, b_{k-1}\}) \\
&+ \psi(B \cup \{b_1, \ldots, b_{k-1}\}) - \psi(B \cup \{b_1, \ldots, b_{k-2}\}) \\
&+ \ldots \\
&+ \psi(B \cup \{b_1\}) - \psi(B) \\
&\leq \psi((A \cap B) \cup \{b_1, \ldots, b_k\}) - \psi((A \cap B) \cup \{b_1, \ldots, b_{k-1}\}) \\
&+ \psi((A \cap B) \cup \{b_1, \ldots, b_{k-1}\}) - \psi((A \cap B) \cup \{b_1, \ldots, b_{k-2}\}) \\
&+ \ldots \\
&+ \psi((A \cap B) \cup \{b_1\}) - \psi(A \cap B) \\
&= \psi((A \cap B) \cup \{b_1, \ldots, b_k\}) - \psi(A \cap B) \\
&= \psi(A) - \psi(A \cap B)
\end{aligned}$$

which gives the desired submodular inequality. ■

Just like submodular functions, we can talk about supermodular functions. A function $\psi : \mathcal{P}(X) \to \mathbb{R}$ is supermodular if and only if $-\psi$ submdoular. In other words, for every $A \subseteq B$ and $b \notin B$,

$$\psi(B \cup \{b\}) - \psi(B) \geq \psi(A \cup \{b\}) - \psi(A)$$

So, for supermodular functions, marginal contributions increase for larger sets. A function $\psi : \mathcal{P}(X) \to \mathbb{R}$ is **modular** if it is both submodular and supermodular (i.e., the inequalities in the definitions of submodular and supermodular functions hold with equality). A useful characterization of modular function essentially says that they are "linear" functions. We skip the proof.

THEOREM **3** *A function $\psi : \mathcal{P}(X) \to \mathbb{R}$ is modular if and only if there is some vector $v : X \to \mathbb{R}$ such that $\psi(S) = \sum_{x \in S} v(x)$ for all $S \subseteq N$.*

2.2 Polymatroid Optimization

For simplicity of notation, we will denote $X = \{x_1, \ldots, x_m\}$. Sometimes, we will also refer to x_i as i. We will be interested in the following polyhedron associated with a polymatroid set function $\psi : P(X) \to \mathbb{R}$: [1]

$$P^\psi := \{z \in \mathbb{R}^m_+ : \sum_{j \in S} z_j \leq \psi(S) \ \forall \ S \subseteq X\}.$$

Technically, z is a map from X to $\mathbb{R}_+$. However, it is easier to write it as a vector in $\mathbb{R}^m_+$, and denote by z_j the value of the map for type x_j.

We will attach a linear program with the feasible set being this polyhedron. In particular, we will consider the following linear program:

$$\max_z \sum_{j \in X} c_j z_j \quad \text{subject to} \quad z \in P^\psi$$

where $c \in \mathbb{R}^{|X|}$. We will denote this linear program as $\mathbf{LP}^\psi$.

$$\begin{aligned} &\max_z \sum_{j \in X} c_j z_j \\ &\text{subject to} \qquad\qquad\qquad\qquad\qquad\qquad (\mathbf{LP}^\psi) \\ &\quad \sum_{j \in S} z_j \leq \psi(S) \qquad \forall \ S \subseteq X \\ &\quad z_j \geq 0 \qquad \forall \ j \in X \end{aligned}$$

We show that a greedy algorithm gives an optimal solution of ($\mathbf{LP}^\psi$). The greedy algorithm works as follows.

1. Without loss of generality, order the coefficients $c_1 \geq c_2 \geq \ldots \geq c_r > 0 \geq c_{r+1} \geq \ldots \geq$

[1] Since $\psi(\emptyset) = 0$ and ψ in non-decreasing, $\psi(S) \geq 0$ for all S.

c_m.

2. Set

$$z_i = \begin{cases} \psi(S^i) - \psi(S^{i-1}) & \text{if } i \in \{1, \ldots, r\} \\ 0 & \text{otherwise} \end{cases}$$

where $S^i = \{1, \ldots, i\}$ for all $i \in \{1, \ldots, r\}$ and $S^0 = \emptyset$.

THEOREM **4** *The greedy algorithm finds an optimal solution of (*$\mathbf{LP}^\psi$*) for any polymatroid set function* $\psi : \mathcal{P}(X) \to \mathbb{R}$.

Proof: As ψ is non-decreasing, $z_i = \psi(S^i) - \psi(S^{i-1}) \geq 0$ for all $i \in \{1, \ldots, r\}$. Also, for each $T \subseteq X$, we note that

$$\begin{aligned} \sum_{j \in T} z_j &= \sum_{j \in T \cap S^r} z_j = \sum_{j \in T \cap S^r} \big[\psi(S^j) - \psi(S^{j-1})\big] \\ &\leq \sum_{j \in T \cap S^r} \big[\psi(S^j \cap T) - \psi(S^{j-1} \cap T)\big] \\ &\leq \sum_{j \in S^r} \big[\psi(S^j \cap T) - \psi(S^{j-1} \cap T)\big] \\ &= \psi(S^r \cap T) - \psi(\emptyset) \\ &\leq \psi(T), \end{aligned}$$

where the first inequality followed from submodularity and the second and the last one from non-decreasingness of ψ.

This shows that the greedy solution is a feasible solution. The objective function value from the greedy solution is

$$\sum_{i=1}^{r} c_i \big[\psi(S^i) - \psi(S^{i-1})\big].$$

Now, to prove optimality, we consider the dual of the linear program $\mathbf{LP}^\psi$. The dual of this

linear program has variables y_S for every $S \subseteq X$.

$$\begin{aligned} \min_{y} \quad & \sum_{S \subseteq X} \psi(S) y_S \\ \text{subject to} \quad & \\ & \sum_{S: j \in S} y_S \geq c_j \qquad \forall\ j \in X \\ & y_S \geq 0 \qquad \forall\ S \subseteq X \end{aligned} \tag{$\mathbf{DLP}^{\psi}$}$$

We first give a solution to the dual ($\mathbf{DLP}^{\psi}$). Let

$$y_S = \begin{cases} c_i - c_{i+1} & \text{if } S = S^i \text{ for some } i \in \{1, \ldots, r-1\} \\ c_r & \text{if } S = S^r \\ 0 & \text{otherwise} \end{cases}$$

Note that non-negativity constraint is satisfied. Further, for any $j \leq r$,

$$\sum_{S: j \in S} y_S \geq y_{S^j} + y_{S^{j+1}} + \ldots + y_{S^r} = c_r + \sum_{i=j}^{r-1} [c_i - c_{i+1}] = c_j$$

For any $j > r$, $c_j \leq 0$, and hence

$$\sum_{S: j \in S} y_S \geq 0 \geq c_j$$

Thus, the $\{y_S\}_S$ is a feasible solution of the dual problem ($\mathbf{DLP}^{\psi}$). The objective function value of the dual from this feasible solution is

$$\sum_{S \subseteq N} \psi(S) y_S = c_r \psi(S^r) + \sum_{i=1}^{r-1} [c_i - c_{i+1}] \psi(S^i) = \sum_{i=1}^{r} c_i \big[\psi(S^i) - \psi(S^{i-1})\big]$$

Hence, the objective function value of the primal feasible solution from the greedy algorithm and the dual feasible solution is the same. By strong duality theorem, these are optimal solutions. ■

Two interesting observations from the proof of Theorem 4 emerge. First, if $\psi(S \cup \{a\}) - \psi(S)$ is integral for all $S \subseteq X$ and $a \in X \setminus S$, then there is a greedy **integral** optimal solution. Second, if $\psi(S \cup \{a\}) - \psi(S) \in \{0, 1\}$ for all $S \subseteq X$ and $a \in X \setminus S$, then the greedy algorithm gives a $\{0, 1\}$ feasible solution that is optimal. In this case, ψ is called a **submodular rank function**.

This point further highlights an important point that even though the constraint matrix is not *totally unimodular*, a widely applied sufficient condition for existence of integral optimal solution of linear programs, we have a new class of problems where LP relaxation solves the integer program.

The proof also highlights another interesting point. It identifies the extreme points of the polyhedron P^ψ. To define the extreme points, let $S \subseteq X$ be some subset of types in X (S may be empty also). Let $\succ_S$ be a strict ordering of types in S. For simplicity, let $S = \{1, \ldots, \ell\}$ and $1 \succ_S 2 \succ_S \ldots \succ_S \ell$ and for every $i \in \{1, \ldots, \ell\}$, define

$$S^i := \{1, \ldots, i\}$$

Then, consider the following feasible solution of P^ψ:

$$z_i = \begin{cases} \psi(S^i) - \psi(S^{i-1}) & \text{if } i \in S \\ 0 & \text{otherwise} \end{cases}$$

We call this a **priority solution** given by $(S, \succ_S)$. As the proof shows every priority solution is a feasible point in P^ψ. Theorem 4 implies that these are the only extreme points of P^ψ.

THEOREM **5** *Every priority solution is an extreme point of P^ψ. Conversely, every extreme point of P^ψ is a priority solution.*

Proof: We use the following characterization of extreme points of a polyhedron. A feasible solution $z^* \equiv (z_1^*, \ldots, z_m^*) \in P^\psi$ is an extreme point of P^ψ if and only if there exists $c_1, \ldots, c_n$ such that z^* is the unique solution of

$$\max_z \sum_{j \in N} c_j z_j \quad \text{subject to} \quad z \in P^\psi$$

Now, fix a priority solution $(S, \succ_S)$. Without loss of generality let $S = \{1, \ldots, r\}$ and $1 \succ_S 2 \succ_S 3 \succ_S \ldots \succ_S r$. Then, choose c such that $c_1 > c_2 > \ldots > c_r > 0 > c_{r+1} \geq c_{r+2} \geq \ldots \geq c_m$. We argue that the priority solution identified in Theorem 4 (via the greedy algorithm) is the unique optimal solution.

Let z be an optimal solution of ($\mathbf{LP}^{\psi}$). Consider the constructed optimal dual solution of ($\mathbf{DLP}^{\psi}$) in the proof of Theorem 4:

$$y_S = \begin{cases} c_i - c_{i+1} & \text{if } S = S^i \text{ for some } i \in \{1, \ldots, r-1\} \\ c_r & \text{if } S = S^r \\ 0 & \text{otherwise} \end{cases}$$

By Complementary slackness, x is optimal if and only if

$$y_S \Big[\psi(S) - \sum_{j \in S} z_j \Big] = 0 \qquad \forall\ S \subseteq X \qquad (\mathbf{CS-1})$$

$$z_j \Big[\sum_{S: j \in S} y_S - c_j \Big] = 0 \qquad \forall\ j \in X \qquad (\mathbf{CS-2})$$

Now if $c_i \neq 0$ for all $i \in X$ and $c_i \neq c_j$ for all $i, j \in X$ with $c_i, c_j > 0$, we get that $y_S > 0$ if $S = S^i$ for some $i \in \{1, \ldots, r\}$ and $y_S = 0$ otherwise. By ($\mathbf{CS-2}$), $z_j = 0$ for all $j \notin \{1, \ldots, r\}$. Also, by ($\mathbf{CS-1}$):

$$\psi(S^i) = \sum_{j \in S^i} z_j \qquad \forall\ j \in \{1, \ldots, r\}$$

Since $S^i := \{1, \ldots, i\}$, this implies that $x_i = \psi(S^i) - \psi(S^{i-1})$ for all $i \in \{1, \ldots, r\}$ This is the solution identified in the greedy algorithm.

This completes the proof that every priority solution is an extreme point. Finally, for any choice of c, there is some priority solution that is an optimal solution. So, every extreme point is a priority solution. ■

2.3 The extreme points of Border polytope

Theorem 5 can be used to derive extreme points of "Border polytope", which is about the extreme points of reduced-form implementable allocation rules. But, we can say even more. We can identify the allocation rules corresponding to these extreme points. For this define the *hierarchical allocation rules* as follows.

DEFINITION **5** *An allocation rule $q : X \times \mathcal{C} \to [0,1]$ is a* **hierarchical allocation rule** *if there exists $S \subseteq X$ and a strict order over the types in $\succ_S$ such that for all $(x,c) \in X \times \mathcal{C}$,*

$$q(x,c) = \begin{cases} 0 & \text{if } c(x) = 0 \text{ or } x \notin S \\ 1 & \text{if } c(x) > 0 \text{ and } x \succ_s y \ \forall\ y \in S \text{ with } c(y) > 0 \end{cases}$$

Hence, in any heirarchical allocation rule, given by $(S, \succ_S)$, the object is not allocated at a type profile if every type is outside S. Otherwise, the object is equally shared by the agents who have their type equal to the highest type in S according to $\succ_S$ that appear in this type profile.

THEOREM **6** *Suppose q is an allocation rule. Then, there exists a collection of heirarchical allocation rules $q^1, \ldots, q^k$ and non-negative weights $\lambda_1, \ldots, \lambda_k \in [0,1]$ such that $\sum_j \lambda_j = 1$ and for every $x \in X$*

$$Q(x) = \sum_{j=1}^{k} \lambda_j Q^j(x)$$

Proof: Define for every $S \subseteq X$

$$\psi(S) := 1 - \Big[\sum_{x \notin S} F(x)\Big]^n$$

Clearly ψ is non-decreasing. Pick any $S \subseteq T$ and $y \notin T$. Then,

$$\psi(T \cup \{y\}) - \psi(T) = \Big[F(y) + \sum_{x \notin (T \cup \{y\})} F(x)\Big]^n - \Big[\sum_{x \notin (T \cup \{y\})} F(x)\Big]^n$$

Denoting $\sum_{x \notin T \cup \{y\}} F(x)$ as F_T^-, we see

$$\begin{aligned}\psi(T \cup \{y\}) - \psi(T) &= \Big[F(y) + F_T^-\Big]^n - \Big[F_T^-\Big]^n \\ &= F(y)\Big[(F(y) + F_T^-)^{n-1} + (F(y) + F_T^-)^{n-2} F_T + (F(y) + F_T^-)^{n-3}(F_T^-)^2 \\ &\quad + \ldots + (F_T^-)^{n-1}\Big]\end{aligned}$$

Note that F_T^- is decreasing in T, i.e., $F_S^- > F_T^-$. Hence,

$$\psi(T \cup \{y\}) - \psi(T) < \psi(S \cup \{y\}) - \psi(S)$$

This shows that ψ is strictly submodular.

By Theorem 1, an interim allocation rule Q is implementable if and only if

$$\sum_{x \in S} nQ(x)F(x) = \sum_{x \in S} \widetilde{Q}(x) \leq \psi(S) \qquad \forall\ S \subseteq X$$

By Theorem 5, we know that any solution of this can be expressed as a convex combination of priority solutions. Hence, any interim allocation rule Q is implementable if and only if it is a convex combination of allocation rules that generate the priority solutions as interim allocation rules. Hence, we show that every interim allocation rule which is a priority solution can be generated from a hierarchical allocation rule. To do so, fix an interim allocation rule Q which is a priority solution $(S, \succ_S)$. Without loss of generality we let $X = \{x_1, \ldots, x_m\}$ and $S = \{x_1, \ldots, x_r\}$ with $x_1 \succ_S \ldots \succ_S x_r$. Hence,

$$\widetilde{Q}(x_i) = \begin{cases} \psi(S^i) - \psi(S^{i-1}) = \Big[\sum_{x \notin S^{i-1}} F(x)\Big]^n - \Big[\sum_{x \notin S^i} F(x)\Big]^n & \text{if } i \in \{1, \ldots, r\} \\ 0 & \text{otherwise} \end{cases}$$

Now, consider the hierarchical allocation rule with same $(S, \succ_S)$. The probability that a type $x \notin S$ is given the object is zero in this allocation rule. The probability that a type $x_i \in S$ gets the object can be computed as follows.

- The probability that an agent of type x_1 wins the object if there is at least one agent

of type x_1, and this probability is

$$1 - \Big[\sum_{x \notin \{x_1\}} F(x)]^n = \Big[\sum_{x \in X} F(x) \Big]^n - \Big[\sum_{x \notin \{x_1\}} F(x) \Big]^n$$

The expected number of agents of type x_1 is $nF(x_1)$. Hence, the probability that an agent of type x_1 wins the object in this hierarchical allocation rule is

$$\frac{1}{nF(x_1)} \Big[\big(\sum_{x \in X} F(x) \big)^n - \big(\sum_{x \notin \{x_1\}} F(x) \big)^n \Big]$$

- The probability that an agent of type x_2 wins the object if there is no agent of type x_1 and at least one agent of type x_2. This probability is

$$\big(\sum_{x \notin \{x_1\}} F(x) \big)^n - \big(\sum_{x \notin \{x_1, x_2\}} F(x) \big)^n$$

The expected number of agents of type x_2 is $nF(x_2)$. Hence, the probability that an agent of type x_2 wins the object in this hierarchical allocation rule is

$$\frac{1}{nF(x_2)} \Big[\big(\sum_{x \notin \{x_1\}} F(x) \big)^n - \big(\sum_{x \notin \{x_1, x_2\}} F(x) \big)^n \Big]$$

- ... For any $j \in \{1, \ldots, r\}$, the probability that an agent of type x_j wins the object if there is no agent of type $x_1, \ldots, x_{j-1}$ and at least one agent of type x_j. This probability is

$$\big(\sum_{x \notin \{x_1, \ldots, x_{j-1}\}} F(x) \big)^n - \big(\sum_{x \notin \{x_1, x_2, \ldots, x_j\}} F(x) \big)^n$$

The expected number of agents of type x_j is $nF(x_j)$. Hence, the probability that an agent of type x_j wins the object in this hierarchical allocation rule is

$$\frac{1}{nF(x_j)} \Big[\big(\sum_{x \notin \{x_1, \ldots, x_{j-1}\}} F(x) \big)^n - \big(\sum_{x \notin \{x_1, x_2, \ldots, x_j\}} F(x) \big)^n \Big]$$

This is exactly the priority solution $(S, \succ_S)$. ■

2.4 The must-sell case: connection to convex game of Shapley

We remain in the symmetric environment. Then, the Border's condition is for all $S \subseteq X$,

$$n \sum_{x \in S} Q(x)F(x) \leq 1 - \big(1 - \sum_{x \in S} F(x)\big)^n \tag{4}$$

However, if the object is allocated with probability one at every type profile, these inequalities are familiar. We call it the *must-sell* case. In the must sell case an allocation rule q must satisfy for every c,

$$\sum_{x \in X} c(x)q(x,c) = 1$$

Hence, we have

$$n \sum_{x \in X} Q(x)F(x) = \sum_{x \in X} \sum_{c \in \mathcal{C}} c(x)q(x,c)\lambda(c) = \sum_{c \in \mathcal{C}} \lambda(c) \sum_{x \in X} c(x)q(x,c) = \sum_{c \in \mathcal{C}} \lambda(c) = 1$$

As a result, we get LHS of Border's condition (4) for $S = X$ equal to

$$n \sum_{x \in X} Q(x)F(x) = 1 \tag{5}$$

In fact if we rewrite the inequality (4) a bit using (5): for all $S \subseteq X$

$$n \sum_{x \in S} Q(x)F(x) \leq 1 - \big(\sum_{x \notin S} F(x)\big)^n = n \sum_{x \in X} Q(x)F(x) - \big(\sum_{x \notin S} F(x)\big)^n$$
$$\Leftrightarrow \Big(\sum_{x \notin S} F(x)\Big)^n \leq n \sum_{x \notin S} Q(x)F(x)$$

In the must-sell case, the Border's inequality can be rewritten as:

$$\Big(\sum_{x \in S} F(x)\Big)^n \leq n \sum_{x \in S} Q(x)F(x) \qquad \forall\ S \subseteq X$$

with equality holding for $S = X$. Note that for any S

$$\psi(S) = 1 - \Big(\sum_{x \notin S} F(x)\Big)^n = \psi(X) - \Big(\sum_{x \notin S} F(x)\Big)^n$$

Hence,

$$W(S) = \Big(\sum_{x \in S} F(x)\Big)^n = \psi(X) - \psi(X \setminus S) \qquad \forall\ S \subseteq X$$

Since ψ is submodular and increasing, we get W is supermodular and increasing.

So, we can define a cooperative game with set of players equal to X, worth of coalition S as

$$W(S) = \Big[\sum_{x \in S} F(x)\Big]^n$$

Then, defining a new variable $\widetilde{Q}(x) = nQ(x)F(x)$ for all $x \in X$, we get the following inequality

$$W(S) \leq \sum_{x \in S} \widetilde{Q}(x) \qquad \forall\ S \subseteq X$$

with equality holding for $S = X$. This is exactly the core inequality for game (X, W). Indeed W is supermodular, and (X, W) is a convex game.

The extreme points of the convex games are well understood. Let $|X| = m$. Then, order the types in X as $x_1 \succ x_2 \ldots \succ x_m$. Now, the following is an extreme point of the convex game:

$$\begin{aligned}
\widetilde{Q}(x_1) &= W(X) - W(X \setminus \{x_1\}) \\
\widetilde{Q}(x_2) &= W(X \setminus \{x_1\}) - W(X \setminus \{x_1, x_2\}) \\
&\ldots\ldots \\
\widetilde{Q}(x_m) &= W(x_m)
\end{aligned}$$

Here, it is clear that $\widetilde{Q}(x_1) \geq \widetilde{Q}(x_2) \geq \ldots \geq \widetilde{Q}(x_m)$.

Example. Suppose $X = \{x_1, x_2, x_3\}$. Assume $F(x_1) = F(x_2) = F(x_3) = \frac{1}{3}$ and three agents $n = 3$. Then, in the must-sell case, we have the following constraints.

$$\begin{aligned} W(\{x_1\}) = W(\{x_2\}) = W(\{x_3\}) &= \left(\frac{1}{3}\right)^3 = \frac{1}{27} \\ W(\{x_1, x_2\}) = W(\{x_2, x_3\}) = W(\{x_1, x_3\}) &= \left(\frac{2}{3}\right)^3 = \frac{8}{27} \\ W(\{x_1, x_2, x_3\}) &= 1 \end{aligned}$$

The core constraints of this game are the Border constraints:

$$\begin{aligned} \widetilde{Q}(x_1) &\geq \frac{1}{27} \\ \widetilde{Q}(x_2) &\geq \frac{1}{27} \\ \widetilde{Q}(x_3) &\geq \frac{1}{27} \\ \widetilde{Q}(x_1) + \widetilde{Q}(x_2) &\geq \frac{8}{27} \\ \widetilde{Q}(x_2) + \widetilde{Q}(x_3) &\geq \frac{8}{27} \\ \widetilde{Q}(x_1) + \widetilde{Q}(x_3) &\geq \frac{8}{27} \\ \widetilde{Q}(x_1) + \widetilde{Q}(x_2) + \widetilde{Q}(x_3) &= 1 \end{aligned}$$

In this case, $\widetilde{Q}(x) = Q(x)$ because $nF(x) = 3 \times \frac{1}{3} = 1$. Hence,

$$\begin{aligned} Q(x_1) &\geq \frac{1}{27} \\ Q(x_2) &\geq \frac{1}{27} \\ Q(x_3) &\geq \frac{1}{27} \\ Q(x_1) + Q(x_2) &\geq \frac{8}{27} \\ Q(x_2) + Q(x_3) &\geq \frac{8}{27} \\ Q(x_1) + Q(x_3) &\geq \frac{8}{27} \\ Q(x_1) + Q(x_2) + Q(x_3) &= 1 \end{aligned}$$

These inequalities are shown in Figure 2 in the simplex.

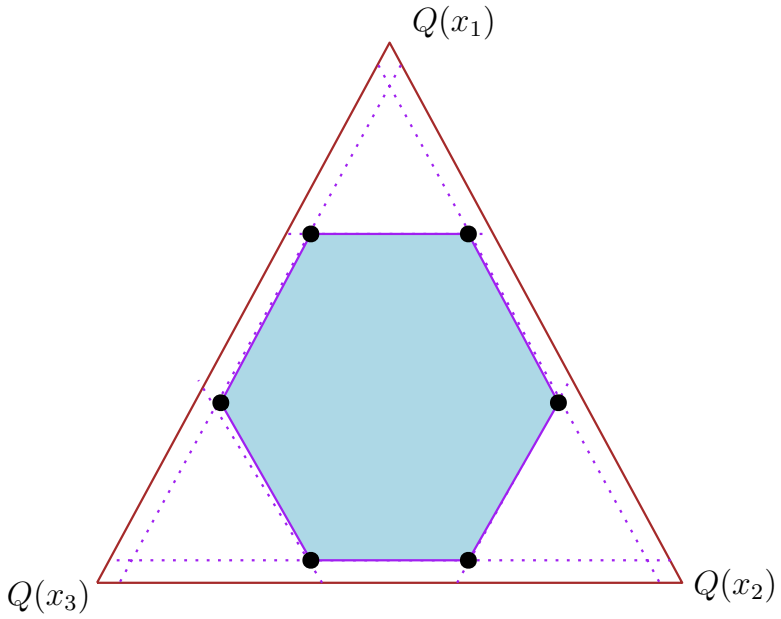

Figure 2: Polytope corresponding to three types

The extreme points are as follows. Assume the order $x_1 \succ x_2 \succ x_3$.

$$Q(x_1) = 3\Big[W(\{x_1, x_2, x_3\}) - W(\{x_2, x_3\})\Big] = 3\frac{19}{81} = \frac{19}{27}$$
$$Q(x_2) = 3\Big[W(\{x_2, x_3\}) - W(\{x_3\})\Big] = 3\frac{7}{81} = \frac{7}{27}$$
$$Q(x_3) = 3W(\{x_3\}) = 3\frac{1}{81} = \frac{1}{27}$$

Note that (at least in this case), independent of the ordering of the types, the first type always gets $\frac{19}{27}$, the second type always gets $\frac{7}{27}$, and the third type always gets $\frac{1}{27}$. Hence, if $x_1, x_2, x_3 \in \mathbb{R}$ and $x_1 > x_2 > x_3$, BIC constraints will imply that $Q(x_1) \geq Q(x_2) \geq Q(x_3)$. And, the only way this is possible to hold at an extreme point is if the ordering of types is $x_1 \succ x_2 \succ x_3$, i.e., the ordering of the types should match the ordering on real number line.

Imposing the monotonicity constraints inside the polytope in Figure 2 gives us Figure 3. Here it is seen that there are exatcly four extreme points: (a) one extreme point is the the *complete bunching* point: $Q(x_1) = \ldots = Q(x_m) = \frac{1}{m}$; (b) the other extreme point is unique monotone extreme point of the original Border's must-sell polytope; (c) the other extrem points are on the *faces* of the original Border's must-sell polytope.

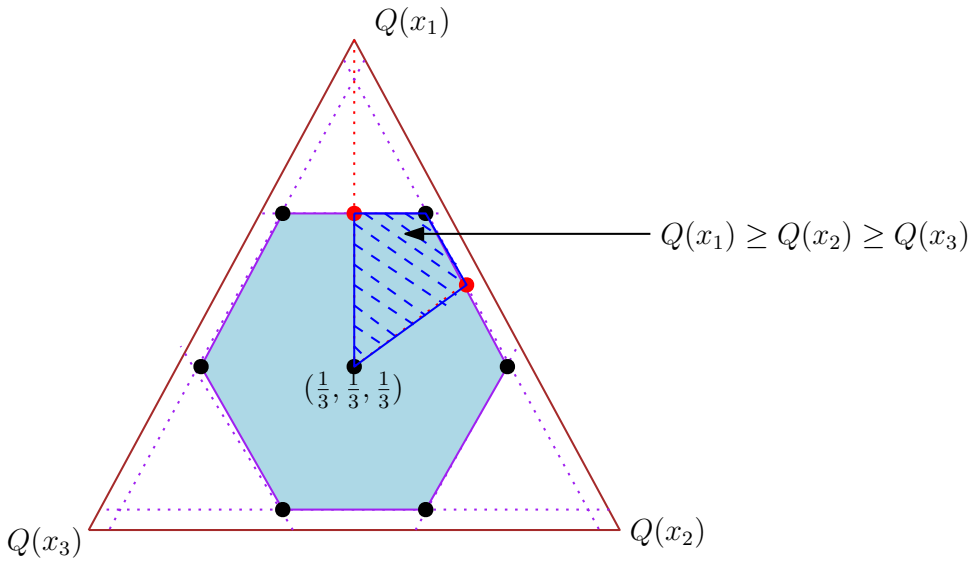

Figure 3: Polytope with monotonicity constraints corresponding to three types

3 Border's theorem: finite asymmmetric case

There is a single indivisible object for sale. There are n agents. The set of agents is denoted by N. For each agent $i \in N$, let V_i be the set of values of agent i for the object. We assume that V_i is finite. Let $V \equiv V_1 \times \ldots \times V_n$.

Each agent i draws her value using a distribution F_i; i.e., for every $v_i \in V_i$, the probability that i has value v_i is equal to $F_i(v_i)$.

An *allocation rule* $q \equiv (q_1, \ldots, q_n)$ is a collection of n maps such that each $q_i : V \to [0,1]$ for every $i \in N$ such that $\sum_{i \in N} q_i(v) \leq 1$ for all $v \in V$. Given an allocation rule q, we can generate the *interim allocation rule* $Q \equiv (Q_1, \ldots, Q_n)$ as follows. For each $i \in N$, the interim allocation rule of agent i is a map $Q_i : V_i \to [0,1]$ such that

$$Q_i(v_i) = \sum_{v_{-i} \in V_{-i}} q_i(v_i, v_{-i}) F_{-i}(v_{-i}) \qquad \forall\ v_i \in V_i$$

where $F_{-i}(v_{-i}) = \prod_{j \neq i} F_j(v_j)$.

Border's thereom asks the following question. If there is an interim allocation rule Q, does there exist an allocation rule q that can generate it? If the answer to the previous question is yes, we say that Q is implementable.

THEOREM **7** *Interim allocation rule Q is implementable if and only if for every $S_1 \subseteq V_1, \ldots, S_n \subseteq V_n$, we have*

$$\sum_{i \in N} \sum_{v_i \in S_i} Q_i(v_i) F_i(v_i) \leq 1 - \prod_{i \in N} \left[1 - \sum_{v_i \in S_i} F_i(v_i) \right]$$

The left hand side is the probability that the object is assigned to a type in $S_1 \cup \ldots \cup S_n$.

3.1 Proof of Border's theorem: network flow

Given an interim allocation rule Q, construct a network as follows:

1. There is a source vertex s and a terminal vertex t.

2. There are two more *layers* of vertices:

(a) *Layer 1.* For every $i \in N$ and for every $v_i \in V_i$, put a vertex for v_i. So, this layer contains a vertex for every type in $V_1 \cup \ldots \cup V_n$

(b) *Layer 2.* For every $v \in V$, there is a vertex corresponding to v.

3. There is a directed edge from source vertex s to every vertex in layer 1, i.e., edges are of the form (s, v_i) for each $i \in N$ and each $v_i \in V_i$. The capacity of this edge is $Q_i(v_i)F_i(v_i)$.

4. There is a directed edge from every vertex $v \in V$ in layer 2 to terminal vertex t. The capacity of this edge is $\prod_{i \in N} F_i(v_i)$.

5. There is a directed edge from each $v_i \in V_i$ to a type profile (v_i, v_{-i}) for every v_{-i}. The capacities of these edges are "sufficiently high".

This fully describes the network. Let $\mathcal{E}$ be the set of edges of this network. Let $\kappa(e)$ denote the capacity of edge $e \in \mathcal{E}$. A flow is a map $x : \mathcal{E} \to \mathbb{R}_+$ such that

1. *capacity constraint:* $x(e) \leq \kappa(e)$ *for all* $e \in \mathcal{E}$.

2. *flow balancing:*

$$x(s, v_i) = \sum_{v_{-i} \in V_{-i}} x(v_i, (v_i, v_{-i})) \qquad \forall\ v_i \in V_i\ \forall\ i \in N$$

$$\sum_{i \in N} x(v_i, v) = x(v, t) \qquad \forall\ v \in V$$

A flow x is **maximal** if for every other flow y

$$\sum_{i \in N} \sum_{v_i \in V_i} x(s, v_i) \geq \sum_{i \in N} \sum_{v_i \in V_i} y(s, v_i)$$

The first step of the proof says Q is implementable if and only if there is a maximum flow where each edge (s, v_i) flows equal to capacity. We show this in two steps.

Necessity. Suppose Q is implementable. Then, it is generated by some allocation rule q. Consider the following candidate flow x:

$$\begin{aligned} x(s, v_i) &= Q_i(v_i)F_i(v_i) = \kappa(s, v_i) \qquad \forall\ i \in N\ \forall\ v_i \in V_i \\ x(v_i, (v_i, v_{-i})) &= q_i(v) \prod_{j \in N} F_j(v) \leq \kappa(v_i, (v_i, v_{-i})) \qquad \forall\ i\ \in N, \forall\ v_i \in V_i,\ \forall\ v_{-i} \in V_{-i} \\ x(v, t) &= \Big(\sum_{i \in N} q_i(v) \Big) \prod_{j \in N} F_j(v) \leq \prod_{j \in N} F_j(v) = \kappa(v, t) \qquad \forall\ v \in V \end{aligned}$$

where all the capacity constraints hold by definition of capacities. This is a flow if flow balancing constraints hold. We verify that below. Pick any vertex v_i in first layer. We see that the sum of outflows

$$\sum_{v_{-i} \in V_{-i}} x(v_i, (v_i, v_{-i})) = F_i(v_i) \sum_{v_{-i} \in V_{-i}} q_i(v_i, v_{-i}) \prod_{j \neq i} F_j(v_j) = F_i(v_i) Q_i(v_i) = x(s, v_i)$$

Next, pick any vertex v in the second layer. We see that the sum of inflows is equal to

$$\sum_{i \in N} x(v_i, v) = \sum_{i \in N} q_i(v) \prod_{j \in N} F_j(v) = \Big(\sum_{i \in N} q_i(v) \Big) \prod_{j \in N} F_j(v) = x(v, t)$$

Hence, flow balancing and capacity constraints hold implying that x is a feasible flow. It is maximal because the flows match capacity on all edges from the source s.

Sufficiency. Suppose x is a maximal flow such that the edges from source to each v_i is flowing at capacity:

$$x(s, v_i) = Q_i(v_i)F_i(v_i) = \kappa(s, v_i) \qquad \forall\ i \in N\ \forall\ v_i \in V_i$$

Then, we can define the allocation rule q as follows. For every $v \in V$ and every $i \in N$,

$$q_i(v) := \frac{x(v_i, (v_i, v_{-i}))}{\prod_{j \in N} F_j(v)}$$

We need to show that this is a feasible allocation rule first. By capacity constraint of

flow from v to t, and the flow balancing constraint at v, we know that

$$\sum_{i \in N} x(v_i, v) = x(v, t) \leq \prod_{j \in N} F_j(v_j)$$

Hence,

$$\sum_{i \in N} q_i(v) = \frac{1}{\prod_{j \in N} F_j(v_j)} \sum_{i \in N} x(v_i, v) \leq 1$$

Finally, we show that Q is implemented by q. The flow balancing constraint of the first layer at vertex v_i means

$$x(s, v_i) = Q_i(v_i) F_i(v_i) = \sum_{v_{-i}} x(v_i, (v_i, v_i)) = \sum_{v_{-i}} q_i(v) \prod_{j \in N} F_j(v)$$
$$\Rightarrow Q_i(v_i) = \sum_{v_{-i}} q_i(v) \prod_{j \neq i} F_j(v)$$

The next step in the proof is the following. For any interim allocation rule Q, the flow in the corresponding network is maximal with saturated (full capacity) edges from the source if and only if Border's inequalities hold. We know that from max-flow min-cut theorem, the capacity of any $s - t$ cut must be at least the maximum flow. Hence, the capacity of any $s - t$ cut must be at least the sum of capacities from s to all first layer vertices:

$$\sum_{i \in N} \sum_{v_i \in V_i} Q_i(v_i) F_i(v_i) \tag{6}$$

In fact this is necessary and sufficient for the max flow to have saturated flows from s to all first layer vertices.

A minimum $s - t$ cut cannot have edges crossing it which go from layer 1 to layer 2 – such edges have sufficiently high capacity. If $S_i \subseteq V_i$ for each i be the subset of vertices considered from the first layer considered in s side of the cut. Let $S = \cup_i S_i$. Then every vertex in layer 2 connected to a vertex in S must also be considered in the s-side – else, the edge between them (which has infinite capacity) will cross the cut. Let T be the set of such

vertices from layer 2. The capacity of such a cut is:

$$\begin{aligned}
&\sum_{i\in N}\sum_{v_i\in V_i\setminus S_i} Q_i(v_i)F_i(v_i) + \sum_{v\in V:\exists i\in N, v_i\in S_i}\prod_{j\in N} F_j(v_j)\\
=&\sum_{i\in N}\sum_{v_i\in V_i\setminus S_i} Q_i(v_i)F_i(v_i) + 1 - \sum_{v:v_j\in V_j\setminus S_j\ \forall\ j}\prod_{j} F_j(v_j)\\
=&\sum_{i\in N}\sum_{v_i\in V_i\setminus S_i} Q_i(v_i)F_i(v_i) + 1 - \prod_{j\in N}\sum_{v_j\in V_j\setminus S_j} F_j(v_j)
\end{aligned}$$

This is at least the capacity of cut in (6) is equivalent to requiring that

$$\sum_{i\in N}\sum_{v_i\in S_i} Q_i(v_i)F_i(v_i) \le 1 - \prod_{i\in N}\Big(1 - \sum_{v_i\in S_i} F_i(v_i)\Big)$$

This proves Border's theorem.

4 The continuous version of Border's theorem