Noname manuscript No.
(will be inserted by the editor)

Mathematical Program Networks

Forrest Laine

Received: date / Accepted: date

Abstract Mathematical Program Networks (MPNs) are introduced in this work. A MPN is a collection of interdependent Mathematical Programs (MPs) which are to be solved simultaneously, while respecting the connectivity pattern of the network defining their relationships. The network structure of an MPN impacts which decision variables each constituent mathematical program can influence, either directly or indirectly via solution graph constraints representing optimal decisions for their decedents. This framework unifies many existing problem formulations, such as Nash Equilibrium problems; bi-, tri-, and multi-level optimization problems, and Equilibrium Programs with Equilibrium Constraints (EPECs). Computing equilibrium points for any of these specific problem formulations would typically require dedicated algorithms and solver code. Within the presented framework, a collection of decision problems can be solved under various network structures, all using the same framework and algorithms. The presented framework not only enables easier exploration of existing equilibrium definitions and problems, but also facilitates the imagining of many other interesting networks of mathematical programs.

Keywords Multilevel · Equilibrium · Mathematical Programming · Bilevel · Trilevel

F. Laine
Tel.: +1-360-305-5642
E-mail: forrest.laine@vanderbilt.edu

1 Basic Notation

For some integer k, let $[k]$ define the index set $\{1, \dots, k\}$. For some vector $\mathbf{x} \in \mathbb{R}^n$, x_j indicates the jth element of $\mathbf{x}$. For some index set $J \subset [n]$, let $[x_j]_{j \in J}$ denote the vector comprised of the elements of $\mathbf{x}$ selected by the indices in J. If $\mathbf{x}^j$, $j \in J$ is some collection of vectors, then $[\mathbf{x}^j]_{j \in J}$ denotes the concatenation of all vectors in that collection. For an index set J, $\mathcal{P}(J)$ is the set of all subsets of J (power set).

For some set $S \subset \mathbb{R}^n$, the closure of S is denoted $\bar{S}$, the complement of S ($\mathbb{R}^n \setminus S$) is denoted S', the interior of S ($\mathbb{R}^n \setminus \overline{(\mathbb{R}^n \setminus S)}$) is Int$S$, and the boundary of S ($\bar{S} \setminus \text{Int}S$) is denoted ∂S. The dual cone of S is denoted $S^* := \{\mathbf{v} \in \mathbb{R}^n : \langle \mathbf{v}, \mathbf{d} \rangle \geq 0 \ \forall \mathbf{d} \in S\}$, where $\langle \cdot, \cdot \rangle$ is the standard inner product on $\mathbb{R}^n$. The convex hull of a set of points $\mathbf{p}^j$, $j \in J$ is denoted conv$(\{\mathbf{p}^j\}_{j \in J})$. The conic hull of a set of vectors $\mathbf{v}^j$, $j \in J$ is denoted coni$(\{\mathbf{p}^j\}_{j \in J})$.

An open ϵ-ball around some point $\mathbf{x} \in \mathbb{R}^n$ is given as $\mathcal{N}_\epsilon(\mathbf{x}) := \{\mathbf{y} \in \mathbb{R}^n : \|\mathbf{y} - \mathbf{x}\| < \epsilon\}$.

2 Formulation

Definition 2.1 (MPN) A Mathematical Program Network (MPN) is defined by the tuple

$$(\{f^i, C^i, J^i\}_{i \in [N]}, E),$$

and represents a directed graph comprised of N decision nodes (mathematical programs). The nodes jointly optimize a vector of decision variables $\mathbf{x} \in \mathbb{R}^n$. Each decision node $i \in [N]$ is represented by a cost function $f^i : \mathbb{R}^n \to \mathbb{R}$, a feasible set $C^i \subset \mathbb{R}^n$, and a decision index set $J^i \subset [n]$. The network structure of the MPN is defined in terms of the set of directed edges $E \subset [N] \times [N]$. The edge $(i, j) \in E$ iff node j is a child of node i.

For ease of notation, some additional sets and vectors are defined. Let n^i be the cardinality of J^i. Then the sub-vectors $\mathbf{x}^i$ of $\mathbf{x}$ are said to be the private decision variables for node i:

$$\mathbf{x}^i \in \mathbb{R}^{n^i} \quad := [x_j]_{j \in J^i}.$$

The set of reachable transitions for an MPN is represented as $R \subset [N] \times [N]$. Specifically, $(i, j) \in R$ iff there exists a path from node i to node j by traversing edges in E. The set of reachable transitions is used to define the following useful sets:

$$\begin{aligned} D^i &:= \{i\} \cup \{j : (i, j) \in R\} \\ D^{-i} &:= [N] \setminus D^i \end{aligned}$$

The vector $\mathbf{x}$ can be indexed according to these sets:

$$\begin{aligned} \mathbf{x}^{D^i} &:= [\mathbf{x}^j]_{j \in D^i} \\ \mathbf{x}^{D^{-i}} &:= [\mathbf{x}^j]_{j \notin D^i}, \end{aligned}$$

The vector $\mathbf{x}^{D^i}$ is the concatenation of all private decision variables for node i and its descendants. The vector $\mathbf{x}^{D^{-i}}$ are all other decision variables. These sub-vectors are often referenced in conjunction with the following shorthand. For any two vectors $\mathbf{x}$ and $\mathbf{x}^*$, the following identity holds:

$$\left(\mathbf{x}^{D^i}, (\mathbf{x}^*)^{D^{-i}}\right) \equiv \mathbf{y} \in \mathbb{R}^n,$$

where

$$y_j := \begin{cases} x_j : & \exists k \in D^i, j \in J^k \\ x_j^* : & \text{else.} \end{cases}$$

This identity can be thought of as a way to combine $\mathbf{x}^{D^i}$ and $(\mathbf{x}^*)^{D^{-i}}$ into a single vector comparable to $\mathbf{x}$ or $\mathbf{x}^*$. Using all of the above-defined terms, the defining object of a MPN node, the solution graph, can be defined. The solution graph for some node $i \in [N]$ is the following:

$$S^i := \left\{ \begin{array}{lll} \mathbf{x}^* \in \mathbb{R}^n \ : \ (\mathbf{x}^*)^{D^i} \in \underset{\mathbf{x}^{D^i}}{\operatorname{argmin}} & f^i\left(\mathbf{x}^{D^i}, (\mathbf{x}^*)^{D^{-i}}\right) \\ \qquad\qquad\qquad\qquad \text{s.t.} & \left(\mathbf{x}^{D^i}, (\mathbf{x}^*)^{D^{-i}}\right) \in C^i \\ & \left(\mathbf{x}^{D^i}, (\mathbf{x}^*)^{D^{-i}}\right) \in S^j, \ (i,j) \in E \end{array} \right\} \tag{1}$$

The argmin in eq. (1) is taken in a local sense, so that S^i defines the set of all $\mathbf{x}$ for which $\mathbf{x}^{D^i}$ are local minimizers to the optimization problem parameterized by $\mathbf{x}^{D^{-i}}$. The role that network architecture plays in each node's optimization problem is made clear from eq. (1). The private decision variables of child nodes are assumed by the parent node(s), with the requirement that the parents are constrained by the solution graphs of the children.

Definition 2.2 (Equilibrium) A point $\mathbf{x}^* \in \mathbb{R}^n$ is an Equilibrium of a MPN iff x^* is an element of each node's solution graph. Specifically, $\mathbf{x}^*$ in an equilibrium iff $\mathbf{x}^* \in S^*$,where

$$S^* := \bigcap_{i \in [N]} S^i. \tag{2}$$

The solution graph of any mathematical program is a subset of its feasible set. Therefore, $S^i \subset S^j \ \forall (i,j) \in E$, therefore $S^i \subset S^j \ \forall (i,j) \in R$, and the following holds:

Corollary 2.1 *Let $I^{\text{source}} \subset [N]$ be any index set such that $\forall j \in [N], \exists i \in I^{\text{source}} : (i,j) \in R$. Then*

$$S^* = \bigcap_{i \in I^{\text{source}}} S^i. \tag{3}$$

A *cyclic network* is a network for which $(i,i) \in R$ for some $i \in [N]$. An *acyclic network* is then a network which is not cyclic. For acyclic MPNs, the index sets in corollary 2.1 correspond to the set of all source node indices, where a source node is a node without any incoming edges. Cyclic MPNs are considered to be degenerate for the following reason.

Corollary 2.2 *Let $I^{\text{conn}} \subset [N]$ be a set of fully-connected nodes in an MPN, meaning $(i,j) \in R$, $\forall i,j \in I^{\text{conn}}$. Let $I^{\text{downstream}} := \{j \notin I^{\text{conn}} : \exists i \in I^{\text{conn}} \text{ s.t. } (i,j) \in R\}$ be the set of all nodes which are descendants of the nodes in I^{conn} (but not themselves in I^{conn}). Then any singleton set of the form*

$$S^{\text{conn}} = \{\mathbf{x}\},\ \mathbf{x} \in \left(\bigcap_{j \in I^{\text{conn}}} C^j \right) \cap \left(\bigcap_{j \in I^{\text{downstream}}} S^j \right) \tag{4}$$

is a valid solution graph for all $i \in I^{\text{conn}}$.

The result of corollary 2.2 is essentially that solution graphs (and therefore equilibrium points) are not meaningfully defined on cyclic network configurations. The cyclic nature of the network translates to a cyclic definition in what it means to be a solution graph, and the resulting circular definition allows for degenerate graphs which dilute the function of the decision nodes. It is for this reason that only acyclic networks are considered in the remainder of this work.

2.1 Quadratic Program Networks

Particular emphasis in this article is given to a special case of MPNs which possess desirable properties from an analysis and computation perspective. These special MPNs are termed Quadratic Program Networks, defined as the following:

Definition 2.3 (QPNs) A Quadratic Program Network (QPN) is a MPN in which the cost function f^i for each decision node is a quadratic function that is convex with respect to the variables $\mathbf{x}^{D^i}$, and the feasible region C^i is a convex (not-necessarily closed) polyhedral region. In other words, each node, when considered in isolation, is a quadratic program.

In the above definition, a not-necessarily closed polyhedral region is defined as a finite intersection of not-necessarily closed halfspaces, i.e. sets of the form

$$H \subset \mathbb{R}^n := \{\mathbf{x} : \langle \mathbf{a}, \mathbf{x} \rangle \gneq b\}, \tag{5}$$

where, $\mathbf{a} \in \mathbb{R}^n$, $b \in \mathbb{R}$, and $\gneq \in \{>, \geq\}$ is used to indicate that the halfspace may or may not be a closed set. Therefore a not-necessarily convex polyhedral region P can be expressed as

$$P \subset \mathbb{R}^n := \bigcap_{i \in [m]} H^i. \tag{6}$$

It will be seen in later sections that QPNs are an important subclass of MPN, since the solution graphs of their constituent nodes are always unions of convex polyhedral regions. This property lends itself to computation, whereas the the solution graphs of general MPNs can quickly become overwhelmingly complex. Even though the QPN framework imposes considerable restrictions on the type of nodes allowed, they retain a remarkable degree of modeling flexibility. This is demonstrated in the following section.

3 Examples

Before continuing with the development of algorithms for solving MPNs (QPNs in particular), some examples are presented with the aim of highlighting how the MPN framework generalizes other well-known problems, as well as the expressivity of the framework from a modeling perspective. An in-depth case study is presented as well, illustrating how network structure affects the equilibrium points between decision nodes. These examples are presented here in the aim of helping motivate MPNs and QPNs as useful problem instances worthy of study.

3.1 Common Equilibrium Problem Formulations

Many common equilibrium problems can be seen to be special cases of the MPN framework. These include the following:

1. Generalized Nash Equilibrium Problems [14,8], and therefore Cournot Competitions [6], are simply MPNs with multiple nodes and no edges connecting them.
2. Bilevel Optimization Problems [5], and therefore Stackelberg Competitions [22], are MPNs with two nodes and an edge from the top-level program or “leader” pointing to the low-level program or “follower”.
3. Multi-leader Multi-follower games [17,19,11] are MPNs with multiple nodes existing on two levels, with nodes in the top level generally having edges directed towards nodes in the bottom level.

3.2 Modeling Motifs in MPNs

Many interesting relationships between decision variables can be modeled through appropriate construction of nodes and network structure in the MPN framework. Some examples of these *motifs* are listed below. Although most of the following examples can be espressed using relatively small networks with only a few nodes, these modeling motifs are composable within the MPN framework, meaning the following examples can appear as subnetworks in a larger MPN.

1. **Min and Max Expressions**
   Consider a mathematical program of the following form:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & \left(\max_{i\in[M]} g_i(\mathbf{x})\right) \geq 0, \end{aligned} \tag{7}$$

   where $g_i : \mathbb{R}^n \to \mathbb{R}, i \in [M]$ are some given functions. Even when the functions g_i are smooth, the constraint in eq. (7) is generally difficult to encode in standard smooth optimization frameworks, since the requirement that just one of the expressions $g_i(\mathbf{x})$ must be non-negative implies a sort of discrete logic. This constraint can be encoded naturally in the MPN framework, as described in the following table.

node	cost	feasible set	priv. vars	child nodes
1	$f(\mathbf{x}^1)$	$\{\mathbf{x} : \mathbf{x}^2 \geq 0\}$	$\mathbf{x}^1$	$\{2\}$
2	$\mathbf{x}^2$	$\{\mathbf{x} : \mathbf{x}^2 \geq g_i(\mathbf{x}^1), \forall i \in [M]\}$	$\mathbf{x}^2$	$\emptyset$

   The decision node 2 attempts to minimize the variable $\mathbf{x}^2$, while constrained to be greater than or equal to $g_i(\mathbf{x}^1)$, for all $i \in [M]$. In other words, $\mathbf{x}^2$ resolves to the maximum of the expressions $g_i(\mathbf{x}^1)$. With this node as a child of node 1, the constraint appearing in (7) is simplified to simply be $\mathbf{x}^2 \geq 0$. At equilibrium, the variable $\mathbf{x}^1$ is a local optimum of (7).
2. **Shared Variables**
   Many mathematical games are posed using the concept of a shared decision variable, as in the following. Consider N programs, defined for each $i \in [N]$ as:

$$\begin{aligned} \min_{\mathbf{x}^i\in\mathbb{R}^{n^i}, \mathbf{y}\in\mathbb{R}^m} \quad & f^i(\mathbf{x}, \mathbf{y}) \\ \text{s.t.} \quad & (\mathbf{x}, \mathbf{y}) \in C^i, \\ & (\mathbf{x}, \mathbf{y}) \in K. \end{aligned} \tag{8}$$

   Here all N players share control of the decision variable $\mathbf{y}$ and respect the constraint $(\mathbf{x}, \mathbf{y}) \in K$. When the set K can be expressed in the form

$$K := \{(\mathbf{x}, \mathbf{y}) : \mathbf{y} = h(\mathbf{x})\} \tag{9}$$

   for some function $h : \mathbb{R}^n \to \mathbb{R}^m$, the variable $\mathbf{y}$ and associated shared constraint can be eliminated from the problems (8) via substitution. However in some cases, it is not desirable to eliminate the variable $\mathbf{y}$, as it can permit a simpler representation of the problems (8). In other cases, the set K may not have such a form (e.g. $K := \{(\mathbf{x}, \mathbf{y}) : \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} = \mathbf{c}\}$ with $\text{rank}(\mathbf{B}) < m$). Therefore the variable $\mathbf{y}$ is not uniquely defined by a choice of $\mathbf{x}$, and substitution is not possible. This implies that the various decision nodes will generally disagree on the optimal value of $\mathbf{y}$, and no equilibrium will exist without further specifying how such disagreements will be resolved. In either case, the presence of the shared variable can be handled by the MPN framework. This is achieved by introducing a decision node responsible for resolving the shared variable:

$$\begin{aligned} \min_{\mathbf{y}\in\mathbb{R}^m} \quad & f^0(\mathbf{x},\mathbf{y}) \\ \text{s.t.} \quad & (\mathbf{x},\mathbf{y}) \in K. \end{aligned} \tag{10}$$

The function f^0 is a modeling choice, and specifies how $\mathbf{y}$ should be decided when it is not uniquely defined by the value of $\mathbf{x}$.

3. **Polyhedral Exclusions**
Consider a mathematical program of the following form:

$$\begin{aligned} \min_{\mathbf{p}\in\mathbb{R}^n} \quad & f(\mathbf{p}) \\ \text{s.t.} \quad & (P^0+\mathbf{p}) \cap P^i = \emptyset,\ i \in [M] \end{aligned} \tag{11}$$

where $P^i \subset \mathbb{R}^n := \{\mathbf{p} : \langle \mathbf{a}^{i,j}, \mathbf{p}\rangle + b^{i,j} \geq \mathbf{0}\ \forall\ j \in [m^i]\},\ i \in \{0\} \cup [M]$ is some collection of polyhedral regions, each defined by m^i vectors $\mathbf{a}^{i,j} \in \mathbb{R}^n$ and offsets $b^{i,j}$, and the summation between P^0 and $\mathbf{p}$ is taken in the Minkowski sense. This program can be interpreted as that of choosing the position of a polyhedron P^0 such as to optimize a cost f, while avoiding intersecting P^0 with any of the polyhedrons $P^i,\ i \in [M]$. The polyhedral intersection constraints are non-differentiable and generally difficult for continuous optimization solvers to handle. However, these expressions can be represented naturally in the MPN framework. First, introduce the variables $\mathbf{d}^i \in \mathbb{R}^n, \mathbf{q}^i \in \mathbb{R}^n, t^i \in \mathbb{R}$, for $i \in [M]$. Now consider mathematical programs of the following form, for each $i \in [M]$:

$$\begin{aligned} \min_{\mathbf{d}^i\in\mathbb{R}^n,\mathbf{q}^i\in\mathbb{R}^n} \quad & \|\mathbf{p}+\mathbf{d}^i-\mathbf{q}^i\|_2 \\ \text{s.t.} \quad & \mathbf{d}^i \in P^0 \\ & \mathbf{q}^i \in P^i \end{aligned} \tag{12}$$

$$\begin{aligned} \min_{t^i\in\mathbb{R}} \quad & -t^i \\ \text{s.t.} \quad & t^i \leq \langle \mathbf{a}^{i,j}, \mathbf{p}+\mathbf{d}^i\rangle + b^{i,j},\ j \in [m^i] \end{aligned} \tag{13}$$

For any value of $\mathbf{p}$, the point $\mathbf{p}+\mathbf{d}^i$ is the closest possible within the translated P^0 to intersecting P^i, and similar to the previous example, the value t^i resolves to be $\min_{j\in[m^i]}\langle \mathbf{a}^{i,j}, \mathbf{p}+\mathbf{d}^i\rangle + b^{i,j}$. If $t^i < 0$, then the point closest to intersecting P^i still lies outside at least one of the intersecting halfspaces which define it. Therefore the solution graphs of the above programs can be used in an MPN framework to to re-express the constraints in the program (11). This MPN would have $1+2M$ nodes: $2M$ for the problems (12) and (13) at each $i \in [M]$ and one parent node similar to (11) but with the constraints replaced with $t^i < 0\ \forall i \in [M]$. This parent node has a directed edges to each of the other nodes.

Note: this example could be accomplished without the nodes (13). Instead, the constraints requiring $t^i < 0$ could be replaced with constraints requiring $\|\mathbf{p}+\mathbf{d}^i-\mathbf{q}^i\| > 0$. Leveraging the nodes (13) allows the same behavior to be captured within the QPN framework (as opposed to relying on non-linear constraints), which allows for solving via the algorithms explored later in this work.

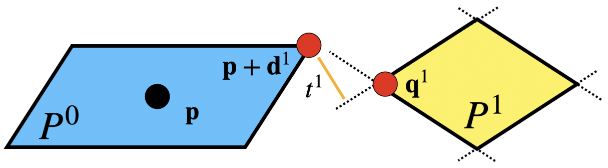

Fig. 1: Depiction of polyhedral exclusion example.

3.3 Constellation Game Case Study

A simple example is presented here to illustrate that the same group of mathematical programs generally have different equilibrium points when considered under different network architectures, and therefore each node in an MPN will have a preference on how the network is arranged. To explore the importance that network architecture has on resulting cost of equilibrium, a small-scale class of randomly generated *Constellation Games* are presented.

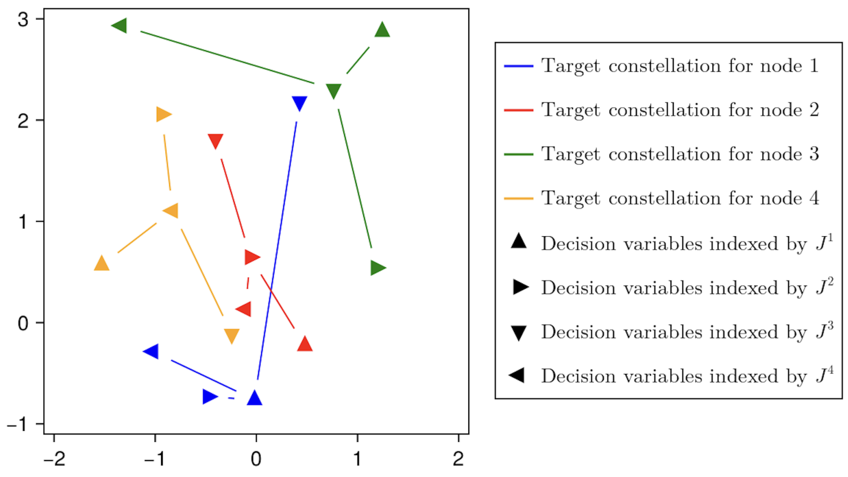

Fig. 2: A depiction of each node's cost function for a randomly generated constellation game. Each constellation (depicted by a color) indicates how a particular node would decide $\mathbf{x}$ if it had unconstrained control over the entire vector. The differently oriented triangles are used to indicate different sub-vectors of $\mathbf{x}$.

The particular example used here is comprised of four MP nodes, indexed 1 through 4. The decision index sets for these MPs are given by

$$J^1 := \{1, 2\}, \quad J^2 := \{3, 4\}, \quad J^3 := \{5, 6\}, \quad J^4 := \{7, 8\},$$

and as such the total decision space dimension is given by $n = 8$. The feasible set for each of the nodes is a box over their respective decision spaces, lifted into $\mathbb{R}^8$:

$$C^i := \{\mathbf{x} \in \mathbb{R}^8 : -5 \leq x_j \leq 5, \quad \forall j \in J^i\}.$$

The cost function for each node is of the following form:

$$f^i(\mathbf{x}) := \|\mathbf{x}^i - \mathbf{g}^i\|_2^2 + \sum_{j=1, j\neq i}^{4} \|\mathbf{x}^j - \mathbf{x}^i - \mathbf{r}^{i,j}\|_2^2,$$

where the vectors $\mathbf{g}^i$ are target locations, and the vectors $\mathbf{r}^{i,j}$ are target relationships between the vectors $\mathbf{x}^i$ and $\mathbf{x}^j$.

In the analysis below, each instance of this class of constellation game is generated by sampling the vectors $\mathbf{g}^i$, $\mathbf{r}^{i,j}$ from the standard unit multivariate normal distribution:

$$\mathbf{g}^i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad \mathbf{r}^{i,j} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}).$$

A depiction of the cost function for each of the four nodes can be seen in fig. 2 for a particular random sampling of the target and relationship vectors.

Using the four nodes so far described, different MPNs can be constructed by relating the nodes with various network architectures. Each of these MPNs correspond to selecting a set of network edges from the super-set of all possible directed edges (excluding self edges):

$$E^{\text{super}} := \{(i,j) : i \in [4], j \in [4], i \neq j\}$$

There are 12 elements in E^{super}, and therefore $2^{12} = 4096$ ways to construct a network architecture for a four-node MPN in this manner. However, for this analysis, only acyclic networks are considered, since cyclic networks result in degenerate MPNs as discussed in section 2. Furthermore, some sets of edges can contain redundant edges, meaning the set of reachable transitions $\mathbf{R}$ for a network is unchanged if those edges are removed. Edge sets with redundant edges are also not considered in this analysis.

Due to symmetry of the nodes in this game, many of the remaining MPNs can also be removed from consideration. From the perspective of any particular node, the other three nodes are interchangeable under a permutation of decision indices. For example, from the perspective of node 1, the networks defined by edge sets $\{(1,2),(2,3)\}$ and $\{(1,4),(4,2)\}$ are identical (until the goal and relationship vectors are realized). Only network architectures in which node 1 is uniquely oriented with respect to its interchangable peers are included.

After removing the cyclic and redundant configurations as defined, there are only 47 remaining configurations to be analyzed. These network configurations are shown in fig. 3, with the yellow-colored node indicating the location of node 1. As stated, the impact that various network architectures have on the yellow node can be used to understand the impact on all other nodes in the constellation game as well via a symmetry argument.

Having these 47 unique network configurations identified, a randomized analysis of the benefit that each architecture provides for a given node was performed. To accomplish this, multiple instances of the constellation game were randomly sampled. For each instance, and for each of the 47 network configurations, an equilibrium was computed (using the methods described in TODO section), and the cost incurred for node 1 at this equilibrium point was logged. Aggregating over 50,000 random game instances, the average cost reduction compared to the Nash equilibrium (the empty edge set) for each of the other configurations was computed, along with the standard error, from which 95% confidence intervals on the average cost estimate could be generated. These cost estimates are displayed for each of the network configurations in fig. 3.

The resulting ordering of network architectures, from most to least advantageous for a given node, is rather fascinating. The Nash configuration with no edges is the worst (all other configurations provide an advantage in comparison). The best configuration, with 6.24% reduction in average cost relative to the Nash configuration is the four-level network with node 1 as the single source node. Interestingly, a four-level network with node 1 appearing further down the hierarchy is still significantly better than many other configurations in which node 1 is a source node, sometimes even the only source node. For example, the edge configuration $\{(2,3),(3,4),(4,1)\}$ (7th best) has a 5.19% cost reduction, while the configuration $\{(1,2),(1,3),(1,4)\}$ (11th best) has a 4.64% reduction.

One conclusion that could be made from these results is that a hierarchical network configuration is advantageous for all nodes in an MPN, even for the nodes at the bottom of the hierarchy. Seemingly it is better to be at the bottom of a strongly hierarchical configuration than it is to be at the top of a configuration in which a clear hierarchy is not established. This is at least true for the constellation game example explored here. Understanding the role that network configuration has on all MPNs is a topic that should be explored in later work.

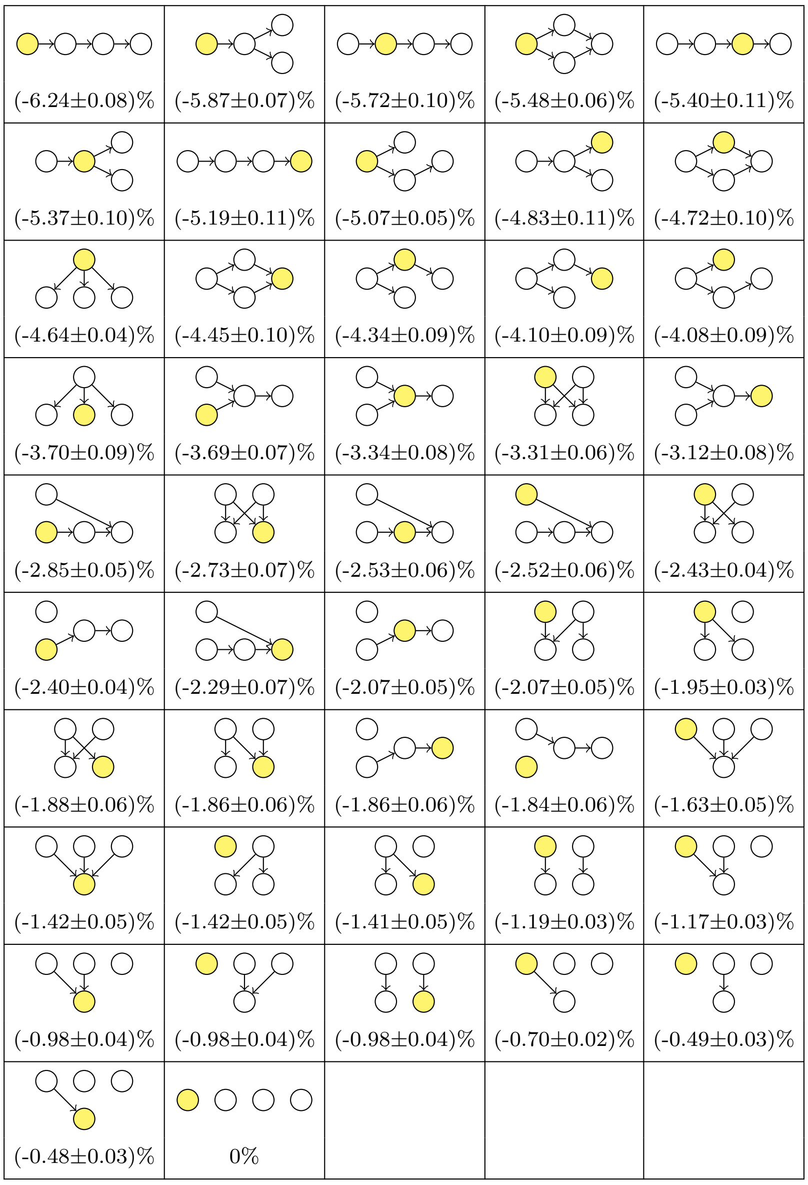

Fig. 3: The 47 unique network configurations for the constellation game described in section 3.3, arranged in order of best to worst for the yellow-colored node. Equilibrium points were found for each configuration, for each of 50,000 random instances of the game. The average relative cost reduction (for the yellow node) compared to the Nash equilibrium (the edgeless configuration) and associated 95% confidence interval are given for each network configuration.

4 Development

In this section, the focus turns towards deriving results which will lead to algorithms for computing equilibrium points for MPNs. The development is centered around the representation of the solution graphs of each node, which are central to the definition of equilibrium (definition 2.2).

Specifically, it is shown that each solution graph is a finite union of convex polyhedral regions. Starting with the childless nodes in the (acyclic) QPN, optimality conditions of optimality lead to a representation of solution graph for those nodes. These representations can then be used to define optimality conditions for those nodes' parents, leading in turn to representations of their solution graphs. An inductive argument leads to representations of the solution graph for all nodes in the network, and therefore a representation for the set of equilibrium points.

Consider a generic parametric optimization problem and its associated solution graph,

$$S := \left\{ \begin{aligned} (\mathbf{x}^*, \mathbf{w}) : \; \mathbf{x}^* \in \underset{\mathbf{x}}{\operatorname{argmin}} \quad & f(\mathbf{x}, \mathbf{w}) \\ \text{s.t.} \quad & (\mathbf{x}, \mathbf{w}) \in C \end{aligned} \right\} \tag{14}$$

where $\mathbf{w} \in \mathbb{R}^m$ is some parameter vector, $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ is a continuous cost function and $C \subset \mathbb{R}^n \times \mathbb{R}^m$ defines a feasible region. For a given $\mathbf{w}$, let $C_\mathbf{w}$ denote the slice of C at $\mathbf{w}$, i.e. $C_\mathbf{w} := \{\mathbf{x} \in \mathbb{R}^n : (\mathbf{x}, \mathbf{w}) \in C\}$.

Definition 4.1 For a given $\mathbf{w}$, a point $\mathbf{x}^*$ with $(\mathbf{x}^*, \mathbf{w}) \in C$ is a local optimum for (14), i.e. $(\mathbf{x}^*, \mathbf{w}) \in S$, if there exists some $\epsilon > 0$ such that

$$f(\mathbf{x}^*, \mathbf{w}) \le f(\mathbf{x}, \mathbf{w}) \; \forall \mathbf{x} \in \mathcal{N}_\epsilon(\mathbf{x}^*) \cap C_\mathbf{w}. \tag{15}$$

Lemma 4.1 *Define $\hat{S}$ as the following:*

$$\hat{S} := \left\{ \begin{aligned} (\mathbf{x}^*, \mathbf{w}) : \; \mathbf{x}^* \in \underset{\mathbf{x}}{\operatorname{argmin}} \quad & f(\mathbf{x}, \mathbf{w}) \\ \text{s.t.} \quad & (\mathbf{x}, \mathbf{w}) \in \overline{C} \end{aligned} \right\}. \tag{16}$$

Then

$$S = \hat{S} \cap C. \tag{17}$$

Recall $\overline{C}$ is the closure of C.

Proof Let $\overline{C}_\mathbf{w}$ be defined analogously to $C_\mathbf{w}$, i.e. the slice of $\overline{C}$ at $\mathbf{w}$. If $(\mathbf{x}^*, \mathbf{w}) \in \hat{S} \cap C$, then

$$f(\mathbf{x}^*, \mathbf{w}) \le f(\mathbf{x}, \mathbf{w}) \; \forall \mathbf{x} \in \mathcal{N}_\epsilon(\mathbf{x}^*) \cap \overline{C}_\mathbf{w}, \tag{18}$$

and since $C_\mathbf{w} \subset \overline{C}_\mathbf{w}$, it is clear that $\hat{S} \cap C \subset S$. For the reverse implication, if $(\mathbf{x}^*, \mathbf{w}) \in S$, then $(\mathbf{x}^*, \mathbf{w}) \in C$, so it must be shown that $S \subset \hat{S}$. This is proved via contradiction.

For a given $\mathbf{w}$, suppose there exists some $\mathbf{x}^*$ such that $(\mathbf{x}^*, \mathbf{w}) \in S$, $(\mathbf{x}^*, \mathbf{w}) \notin \hat{S}$. Assume without loss of generality that $f(\mathbf{x}^*, \mathbf{w}) = 0$. This implies the existence of some $\epsilon > 0$ and $\hat{\mathbf{x}}$ such that $f(\hat{\mathbf{x}}, \mathbf{w}) < 0$, with $\hat{\mathbf{x}} \in \mathcal{N}_\epsilon(\mathbf{x}^*) \cap (\overline{C}_\mathbf{w} \setminus C_\mathbf{w})$. Note that $(\overline{C}_\mathbf{w} \setminus C_\mathbf{w}) \subset \partial C_\mathbf{w}$. Let $\delta = |f(\hat{\mathbf{x}}, \mathbf{w})|$. By the continuity of f, there exists an open neighborhood of $\hat{\mathbf{x}}$ such that $|f(\mathbf{x}, \mathbf{w}) - f(\hat{\mathbf{x}}, \mathbf{w})| < \delta$ for all points $\mathbf{x}$ in this neighborhood, but since $\hat{\mathbf{x}} \in \partial C_\mathbf{w}$, every open neighborhood of $\hat{\mathbf{x}}$ contains at least one point in $C_\mathbf{w}$. Since $\hat{\mathbf{x}} \in \mathcal{N}_\epsilon(\mathbf{x}^*)$, an open set, all sufficiently local neighborhoods of $\hat{\mathbf{x}}$ are also subsets of $\mathcal{N}_\epsilon(\mathbf{x}^*)$. This implies that there exists at least one point $\mathbf{x} \in \mathcal{N}_\epsilon(\mathbf{x}^*) \cap C_\mathbf{w}$ with $|f(\mathbf{x}, \mathbf{w}) - f(\hat{\mathbf{x}}, \mathbf{w})| < \delta$, i.e. $f(\mathbf{x}, \mathbf{w}) < 0$, which contradicts $\mathbf{x}^*$ being a local optimum for (14). Therefore, the assumption must be incorrect, and $S \subset \hat{S}$ as desired. □

Lemma 4.2 *For arbitrary* $\tilde{\mathbf{x}} \in \mathbb{R}^n$, $\tilde{\mathbf{w}} \in \mathbb{R}^m$, *and* $\epsilon > 0$, *let* $\tilde{S}_\epsilon(\hat{\mathbf{x}}, \hat{\mathbf{w}})$ *be defined as the following:*

$$\tilde{S}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}}) := \left\{ \begin{aligned} (\mathbf{x}^*, \mathbf{w}) : \ \mathbf{x}^* \in \underset{\mathbf{x}}{\operatorname{argmin}} \quad & f(\mathbf{x}, \mathbf{w}) \\ \text{s.t.} \quad & (\mathbf{x}, \mathbf{w}) \in C \cap \mathcal{N}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}}) \end{aligned} \right\}. \tag{19}$$

Then

$$S \cap \mathcal{N}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}}) = \tilde{S}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}}) \tag{20}$$

Proof Consider some $(\mathbf{x}^*, \mathbf{w}) \in \tilde{S}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}})$. Then by definition, $(\mathbf{x}^*, \mathbf{w}) \in C \cap \mathcal{N}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}})$, and there exists some $\epsilon' > 0$ such that

$$f(\mathbf{x}^*, \mathbf{w}) \leq f(\mathbf{x}, \mathbf{w}) \ \forall \mathbf{x} \in \mathcal{N}_{\epsilon'}(\mathbf{x}^*) \cap C_\mathbf{w} \cap (\mathcal{N}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}}))_\mathbf{w}, \tag{21}$$

since $(C \cap \mathcal{N}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}}))_\mathbf{w} = C_\mathbf{w} \cap (\mathcal{N}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}}))_\mathbf{w}$. Because $(\tilde{\mathbf{x}}, \tilde{\mathbf{w}}))_\mathbf{w}$ is an open set containing $\mathbf{x}^*$, there exists a sufficiently small choice of ϵ' such that $\mathcal{N}_{\epsilon'}(\mathbf{x}^*) \subset (\mathcal{N}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}}))_\mathbf{w}$. But this implies

$$f(\mathbf{x}^*, \mathbf{w}) \leq f(\mathbf{x}, \mathbf{w}) \ \forall \mathbf{x} \in \mathcal{N}_{\epsilon'}(\mathbf{x}^*) \cap C_\mathbf{w}, \tag{22}$$

which implies $(\mathbf{x}^*, \mathbf{w}) \in S \cap \mathcal{N}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}})$, and therefore $\tilde{S}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}}) \subset S \cap \mathcal{N}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}})$.

Now, for the reverse implication, consider some $(\mathbf{x}^*, \mathbf{w}) \in S \cap \mathcal{N}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}})$. Then $(\mathbf{x}^*, \mathbf{w}) \in C$, $(\mathbf{x}^*, \mathbf{w}) \in \mathcal{N}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}})$, and there exists some ϵ' such that

$$f(\mathbf{x}^*, \mathbf{w}) \leq f(\mathbf{x}, \mathbf{w}) \ \forall \mathbf{x} \in C_\mathbf{w} \cap \mathcal{N}_{\epsilon'}(\mathbf{x}^*). \tag{23}$$

Since $C_\mathbf{w} \subset C_\mathbf{w} \cap (\mathcal{N}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}}))_\mathbf{w} = (C \cap \mathcal{N}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}}))_\mathbf{w}$, this also implies $(\mathbf{x}^*, \mathbf{w}) \in \tilde{S}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}})$, and $S \cap \mathcal{N}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}}) \subset \tilde{S}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}})$. Therefore,

$$S \cap \mathcal{N}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}}) \subset \tilde{S}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}}) \subset S \cap \mathcal{N}_\epsilon(\tilde{\mathbf{x}}, \tilde{\mathbf{w}}). \tag{24}$$

□

Now, consider a region defined to be the finite union of a number of other regions:

$$U := \bigcup_{j \in J} U^j, \tag{25}$$

with $U^j \subset \mathbb{R}^n \times \mathbb{R}^m$, and $J \subset \mathbb{N}$. Using these sets, define the following programs and associated solution graphs, where f is again a continuous function:

$$R := \left\{ \begin{array}{ll} (\mathbf{x}^*, \mathbf{w}) : \ \mathbf{x}^* \in \underset{\mathbf{x}}{\operatorname{argmin}} & f(\mathbf{x}, \mathbf{w}) \\ \qquad\qquad\qquad \text{s.t.} & (\mathbf{x}, \mathbf{w}) \in U \end{array} \right\} \tag{26}$$

and for each $j \in J$,

$$R^j := \left\{ \begin{array}{ll} (\mathbf{x}^*, \mathbf{w}) : \ \mathbf{x}^* \in \underset{\mathbf{x}}{\operatorname{argmin}} & f(\mathbf{x}, \mathbf{w}) \\ \qquad\qquad\qquad \text{s.t.} & (\mathbf{x}, \mathbf{w}) \in \overline{U^j} \end{array} \right\}. \tag{27}$$

Let $U_{\mathbf{w}}$ and $\overline{U^j}_{\mathbf{w}}$ be defined analogously to $C_{\mathbf{w}}$. For some $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{w} \in \mathbb{R}^m$, define

$$\gamma_U(\mathbf{x}, \mathbf{w}) := \{ j \in J : (\mathbf{x}, \mathbf{w}) \in \overline{U^j} \}. \tag{28}$$

Lemma 4.3 *Given a set U of the form (25), a sufficiently small ϵ, and any $(\mathbf{x}, \mathbf{w})$,*

$$\gamma_U(\hat{\mathbf{x}}, \hat{\mathbf{w}}) \subset \gamma_U(\mathbf{x}, \mathbf{w}), \quad \forall (\hat{\mathbf{x}}, \hat{\mathbf{w}}) \in \mathcal{N}_\epsilon(\mathbf{x}, \mathbf{w}). \tag{29}$$

Proof Note that $(\mathbf{x}, \mathbf{w}) \in (\overline{U^j})' \ \forall j \notin \gamma_U(\mathbf{x}, \mathbf{w})$, and all sets $(\overline{U^j})'$ are open. Therefore for sufficiently small ϵ,

$$(\hat{\mathbf{x}}, \hat{\mathbf{w}}) \in (\overline{U^j})' \ \forall j \notin \gamma_U(\mathbf{x}, \mathbf{w}), \ \forall (\hat{\mathbf{x}}, \hat{\mathbf{w}}) \in \mathcal{N}_\epsilon(\mathbf{x}, \mathbf{w}). \tag{30}$$

This implies the result. □

Lemma 4.4 *Given the definitions of R and R^j in (26) and (27),*

$$(\mathbf{x}^*, \mathbf{w}) \in R \iff \left((\mathbf{x}^*, \mathbf{w}) \in U, \ \textit{and} \ (\mathbf{x}^*, \mathbf{w}) \in R^j \ \forall j \in \gamma_U(\mathbf{x}^*, \mathbf{w}) \right). \tag{31}$$

Proof Applying lemma 4.1, we have that $\mathbf{x}^*$ is a local opt for (26) iff $\mathbf{x}^* \in U_{\mathbf{w}}$ and $\exists \epsilon$:

$$f(\mathbf{x}^*, \mathbf{w}) \leq f(\mathbf{x}, \mathbf{w}) \ \forall \mathbf{x} \in \mathcal{N}_\epsilon(\mathbf{x}^*) \cap \overline{U}_{\mathbf{w}}. \tag{32}$$

Note that $\overline{U}_{\mathbf{w}} = \bigcup_{j \in J} \overline{U}^j_{\mathbf{w}}$. Therefore, by the definition of γ_U, $\exists \epsilon > 0$ such that

$$(\mathbf{x}^*, \mathbf{w}) \in \overline{U^j}, \ f(\mathbf{x}^*, \mathbf{w}) \leq f(\mathbf{x}, \mathbf{w}) \ \forall \mathbf{x} \in \mathcal{N}_\epsilon(\mathbf{x}^*) \cap \overline{U^j}_{\mathbf{w}}, \ \forall j \in \gamma_U(\mathbf{x}^*, \mathbf{w}). \tag{33}$$

This proves the $\Rightarrow$ direction. To prove the $\Leftarrow$ direction, it is already stated that $(\mathbf{x}^*, \mathbf{w}) \in U$, so all that is left to prove is that the condition (33) implies (32). To see this, note that for sufficiently small ϵ,

$$\overline{U} \cap \mathcal{N}_\epsilon(\mathbf{x}^*, \mathbf{w}) = \bigcup_{j \in J} \overline{U^j} \cap \mathcal{N}_\epsilon(\mathbf{x}^*, \mathbf{w}) = \bigcup_{j \in \gamma_U(\mathbf{x}^*, \mathbf{w})} \overline{U^j} \cap \mathcal{N}_\epsilon(\mathbf{x}^*, \mathbf{w}), \tag{34}$$

where the last equality is implied in one direction by the fact that $\gamma_U(\mathbf{x}^*, \mathbf{w}) \subset J$, and in the other direction by lemma 4.3. This further implies that

$$\overline{U}_{\mathbf{w}} \cap \mathcal{N}_\epsilon(\mathbf{x}^*) = \bigcup_{j \in \gamma_U(\mathbf{x}^*, \mathbf{w})} \overline{U^j}_{\mathbf{w}} \cap \mathcal{N}_\epsilon(\mathbf{x}^*), \tag{35}$$

which when taken with (33), results in (32 as desired. □

Corollary 4.1 *Given the definitions of R and R^j in (26) and (27),*

$$R = \bigcup_{\substack{J_1 \in \mathcal{P}(J) \\ J_2 = J \setminus J_1}} \left(U \bigcap_{j_1 \in J_1} R^{j_1} \bigcap_{j_2 \in J_2} \left(\overline{U^{j_2}}\right)' \right). \tag{36}$$

Proof For some $J_1 \subset J$, $J_2 = J \setminus J_1$, and a set of the form

$$\Gamma_U(J_1, J_2) := \bigcap_{j_1 \in J_1} \overline{U^{j_1}} \bigcap_{j_2 \in J_2} \left(\overline{U^{j_2}}\right)', \tag{37}$$

lemma 4.4 states that

$$R \cap \Gamma_U(J_1, J_2) = U \bigcap_{j_1 \in J_1} R^{j_1} \bigcap_{j_2 \in J_2} \left(\overline{U^{j_2}}\right)'. \tag{38}$$

Considering all possible sets $J_1 \subset J$ and $J_2 = J \setminus J_1$ gives the result. $\square$

Corollary 4.2 *Given the definitions of R and R^j in (26) and (27), there exists $\epsilon > 0$ such that*

$$R \cap \mathcal{N}_\epsilon(\mathbf{x}, \mathbf{w}) = \bigcup_{\substack{J_1 \in \mathcal{P}(\gamma_U(\mathbf{x},\mathbf{w})) \\ J_2 = \gamma_U(\mathbf{x},\mathbf{w}) \setminus J_1}} \left(U \bigcap_{j_1 \in J_1} R^{j_1} \bigcap_{j_2 \in J_2} \left(\overline{U^{j_2}}\right)' \bigcap \mathcal{N}_\epsilon(\mathbf{x}, \mathbf{w}) \right). \tag{39}$$

Proof This follows immediately from corollary 4.1 and noting that lemma 4.3 implies that for all $j \notin \gamma_U(\mathbf{x}, \mathbf{w})$, $\mathcal{N}_\epsilon(\mathbf{x}, \mathbf{w}) \bigcap \overline{U^j} = \varnothing$, and therefore since $R^j \subset \overline{U^j}$, $\mathcal{N}_\epsilon(\mathbf{x}, \mathbf{w}) \bigcap R^j = \varnothing$. Furthermore, $\mathcal{N}_\epsilon(\mathbf{x}, \mathbf{w}) \subset (\overline{U^j})'$. Together, it's made clear that the entire set J need not be considered, but rather only the local subset $\gamma_U(\mathbf{x}, \mathbf{w})$. $\square$

Corollary 4.2 states that a local representation of the solution graph (26) can be constructed using local representations of the solution graphs (27). This is significant, since the solution graphs R^j can often be easily represented in cases of practical interest. This fact serves as the foundation for testing whether points are equilibrium solutions for MPNs as well as computing solutions, as described below.

4.1 Quadratic Program Networks

QP Networks have some desirable properties, which when taken together with the results in the preceding section, can be leveraged to derive algorithms for identifying equilibrium points. These properties are summarized in the following claims:

Lemma 4.5 *Consider an optimization problem of the form (14), where f is a quadratic function and convex with respect to $\mathbf{x}$, and C is a closed, non-empty polyhedron in $\mathbb{R}^n \times \mathbb{R}^m$, i.e.*

$$C = \bigcap_{i \in [l]} H^i, \tag{40}$$

where, for each $i \in [l]$, and some $\mathbf{a}^i \in \mathbb{R}^n$, and $\mathbf{b}^i \in \mathbb{R}^m$, $c^i \in \mathbb{R}$,

$$H^i = \left\{ (\mathbf{x}, \mathbf{w}) \in \mathbb{R}^n \times \mathbb{R}^m : \langle \mathbf{x}, \mathbf{a}^i \rangle + \langle \mathbf{w}, \mathbf{b}^i \rangle \geq c^i \right\}. \tag{41}$$

Then the solution graph (14) is a union of polyhedral regions, and is given by

$$S = \bigcup_{L \in \mathcal{P}([l])} \left\{ (\mathbf{x}, \mathbf{w}) \in C : \begin{matrix} (\mathbf{x}, \mathbf{w}) \in \cap_{i \in L} \partial H^i \\ \nabla_x f(\mathbf{x}, \mathbf{w}) \in \text{coni}(\{\mathbf{a}^i\}_{i \in L}) \end{matrix} \right\}. \tag{42}$$

Proof Because the problem (14) is a convex optimization problem with linear constraints, the set of solutions are given by the necessary and sufficient conditions of optimality,

$$S = \{ (\mathbf{x}, \mathbf{w}) \in C : \nabla_{\mathbf{x}} f(\mathbf{x}, \mathbf{w}) \in (C_{\mathbf{w}} - \mathbf{x})^* \}. \tag{43}$$

Note that $C_{\mathbf{w}} - \mathbf{x}$ is the tangent cone to $C_{\mathbf{w}}$ at $\mathbf{x}$. The dual cone of an intersection of sets is given by the Minkowski sum of the dual of each set being intersected. Therefore

$$(C_{\mathbf{w}} - \mathbf{x})^* = \sum_{i=1}^{l} (H^i_{\mathbf{w}} - \mathbf{x})^*, \tag{44}$$

where each expression in the sum is given as

$$(H^i_{\mathbf{w}} - \mathbf{x})^* = \begin{cases} \{\mathbf{0}\}, & \mathbf{x} \in \text{Int} H^i_{\mathbf{w}} \\ \{t \cdot \mathbf{a}^i, \ \forall t \geq 0\}, & \mathbf{x} \in \partial H^i_{\mathbf{w}}. \end{cases} \tag{45}$$

Hence, the only terms contributing to the sum in (44 are those for which $\mathbf{x} \in \text{Int} H^i_{\mathbf{w}}$. Therefore

$$(C_{\mathbf{w}} - \mathbf{x})^* = \left\{ \left(\sum_{i : \mathbf{x} \in \text{Int} H^i_{\mathbf{w}}} t^i \cdot \mathbf{a}^i \right), t^i \geq 0 \right\} = \text{coni}(\{\mathbf{a}^i\}_{i : \mathbf{x} \in \text{Int} H^i_{\mathbf{w}}}). \tag{46}$$

Using the fact that $\mathbf{x} \in \partial H^i_{\mathbf{w}} \iff (\mathbf{x}, \mathbf{w}) \in \partial H^i$, and considering all possible boundaries ∂H^i that $(\mathbf{x}, \mathbf{w})$ can lie within, the result follows. To see that each of the sets in the union (42) are polyhedral, note that the dual cone can be expressed in halfspace representation, i.e. there exist vectors $\{\mathbf{y}^j\}_{j \in J_L}$ such that

$$\mathbf{g} \in \text{coni}(\{\mathbf{a}^i\}_{i \in L}) \iff \langle \mathbf{g}, \mathbf{y}^j \rangle \geq 0, \forall j \in J_L. \tag{47}$$

Since f is quadratic, $\nabla_{\mathbf{x}} f(\mathbf{x},\mathbf{w})$ is an affine expression, i.e. can be expressed as $\mathbf{Qx}+\mathbf{Rw}+\mathbf{q}$, for some matrices $\mathbf{Q}\in\mathbb{R}^{n\times n}, \mathbf{R}\in\mathbb{R}^{n\times m}$, and vector $\mathbf{q}\in\mathbb{R}^n$. Therefore, (42) can equivalently be expressed as

$$S = \bigcup_{L\in\mathcal{P}([l])} \left\{ (\mathbf{x},\mathbf{w})\in C : \begin{array}{ll} \langle \mathbf{x},\mathbf{a}^i\rangle + \langle \mathbf{w},\mathbf{b}^i\rangle = c^i, & \forall i\in L \\ \langle \mathbf{x},\mathbf{Q}^\intercal\mathbf{y}^j\rangle + \langle \mathbf{w},\mathbf{R}^\intercal\mathbf{y}^j\rangle \geq -\langle \mathbf{q},\mathbf{y}^j\rangle, & \forall j\in J_L \end{array} \right\}, \tag{48}$$

which is clearly a union of polyhedral regions. □

Lemma 4.6 *Consider some not-necessarily closed polyhedral region P of the form (6), i.e. the intersection of not-necessarily closed halfspaces H^i, $i\in[m]$. If* $\mathrm{Int}P\neq\emptyset$*, then*

$$\overline{P} = \bigcap_{i\in[m]} \overline{H^i}. \tag{49}$$

Proof As in [16], theorem 6.5.

Lemma 4.7 *The solution graph of any node i in an acyclic QPN can be expressed as a union of not-necessarily closed polyhedral regions, i.e.*

$$S^i = \bigcup_{j\in J_i} P^j, \tag{50}$$

where J_i is some index set, and each set P^j is of the form (6).

Proof Consider first any node j without children. As with all nodes in a QPN, the region C^j is polyhedral and not-necessarily closed. Then lemmas 4.1 and 4.5 state that the solution graph must be given by the intersection of C^j and a set of the form (42), which results in a union of polyhedral regions, which each may not be closed if C^j is not closed.

Now consider any node i which does have children, and assume that the solution graphs of its children are unions of the above form. It will be seen that the solution graph of node i is also of that form. First, note that the decision problem for any such node is of the form (26), where the set U is the intersection of C^i and all solution graphs of its children. Hence U is a union of not-necessarily closed polyhedral regions. Corollary 4.1 states that the solution graph is given by a union of sets formed by intersecting U (union of not-necessarily closed polyhedra), sets of the form R^{j_1} (unions of closed polyhedra), and sets of the form $(\overline{U^{j_2}})'$ (unions of open polyhedra). This intersection gives rise again to unions of not-necessarily closed polyhedra, implying the result. □

Corollary 4.3 *The set of equilibria for a QPN is a union of not-necessarily closed polyhedral regions.*

The preceding results imply computational routines for checking whether a point is an equilibrium for a QPN, and similarly, searching for an equilibrium. These schemes are described in the next section.

5 QPN Equilibrium Computation

The results of the preceding section established that the solution graph for each node in a QPN is a finite union of polyhedral regions. It is easy to see that the number of regions comprising these unions can be exceedingly large. Nevertheless, lemma 4.2 suggests that to check whether some point is an element of any node's solution graph, only those regions local to the point need be considered. This is the key concept that will drive the computational schemes developed in this section.

Throughout this section, it will be assumed that the following routines are available for use:

– A solver capable of efficiently finding a solution to an optimization problem of the form (14) in which f is a convex quadratic function, and C is a closed, non-empty polyhedron. One such freely available solver is [18].
– A computational routine capable of transforming between H- and V- representations of polyhedra. One such freely available routine is [10].

With these foundational routines, an algorithm is first presented for checking whether some point $\mathbf{x}$ is an equilibrium of a given QPN.

Algorithm 1 QPN Equilibrium Check

```
Require: QPN
Require: Candidate $\mathbf{x} \in \mathbb{R}^n$
Require: Reverse topological ordering $T$ over the nodes in the QPN.
 1: for $i \in T$ do
 2:     $U \leftarrow C^i \bigcap_{j:(i,j)\in E} \tilde{S}^j_\epsilon(\mathbf{x})$
 3:     for $U^j \in U$ do
 4:         Compute $R^j_\epsilon(\mathbf{x}) = R^j \cap \mathcal{N}_\epsilon(\mathbf{x})$ (27)
 5:         if $\mathbf{x} \notin R^j_\epsilon(\mathbf{x})$ then
 6:             Return false
 7:         end if
 8:     end for
 9:     Compute $\tilde{S}^i_\epsilon(\mathbf{x}) = R \cap N_\epsilon(\mathbf{x})$ (39)
10: end for
11: Return true
```

In algorithm 1, a reverse topological ordering T is an ordering of node indices, such that index j appears before i if (i, j) is an edge of the graph. On line 4, the computation involves enumerating the polyhedral regions in (42) for which $\mathbf{x}$ is an element. This can be accomplished practically by examining the dual variables associated with each halfspace constraint. One these regions are enumerated, excplicit polyhedral regions in halfspace representation can be found by converting the associated dual cones to halfspace representation as in (47), and then constructing the sets in (48). The computation on line 10 results in a valid local representation of the solution graph for node i, as stated by corollary 4.2. This local representation is sufficient for the computation of solution graphs for all subsequent nodes in the ordering T, as stated by lemma 4.2. The algorithm proceeds to check whether $\mathbf{x}$ is an element of each node in the QPN, and returns **true** if $\mathbf{x}$ is an element of all, which implies $\mathbf{x}$ is an equilibrium point.

Algorithm 1 can be extended to search for equilibrium points of QPNs.

Algorithm 2 QPN Equilibrium Search

```
Require: QPN
Require: Candidate $\mathbf{x} \in \mathbb{R}^n$
Require: Reverse topological ordering $T$ over the nodes in the QPN.
 1: for $i \in T$ do
 2:     $U \leftarrow C^i \bigcap_{j:(i,j)\in E} \tilde{S}^j_\epsilon(\mathbf{x})$
 3:     for $U^j \in U$ do
 4:         Compute $R^j_\epsilon(\mathbf{x}) = R^j \cap \mathcal{N}_\epsilon(\mathbf{x})$ (27)
 5:         if $\mathbf{x} \notin R^j_\epsilon(\mathbf{x})$ then
 6:             Choose $\mathbf{x} \leftarrow\in R^j_\epsilon(\mathbf{x})$
 7:             go to line 1
 8:         end if
 9:     end for
10:     Compute $\tilde{S}^i_\epsilon(\mathbf{x}) = R \cap N_\epsilon(\mathbf{x})$ (39)
11: end for
12: Return true
```

Algorithm 2 makes only a minor modification to algorithm 1. When it is detected that the current iterate $\mathbf{x}$ is not an element of any solution graph for any node, one such point is selected, and the algorithm restarts (with the beginning of the reverse topological ordering). This is required, since after updating the point $\mathbf{x}$, all nodes lower in the topological ordering must re-establish the local pieces of their solution graph, if the computations at their parent nodes are to be accurate.

6 Conclusion

The concept of a Mathematical Program Network was developed. MPNs offer a framework for modeling interactions between multiple decision-makers in a manner which enables easy rearranging of the information structure or depth of reasoning each decision process possesses. Several key results were developed to support algorithms for computing equilibrium points to MPNs, and in particular, Quadratic Program Networks. Some example networks and analyses on their solutions were presented. Future work will consider generalizing the computational routines to the general MPN setting, rather than focusing only on the QPN setting. Furthermore, broadening the framework to account for partial information games will be an intriguing direction to consider.

References

1. Arrow, K.J., Debreu, G.: Existence of an equilibrium for a competitive economy. Econometrica: Journal of the Econometric Society pp. 265–290 (1954)
2. Bard, J.F.: An investigation of the linear three level programming problem. IEEE Transactions on Systems, Man, and Cybernetics (5), 711–717 (1984)
3. Bard, J.F., Falk, J.E.: An explicit solution to the multi-level programming problem. Computers & Operations Research **9**(1), 77–100 (1982)
4. Başar, T., Olsder, G.J.: Dynamic noncooperative game theory. SIAM (1998)
5. Bracken, J., McGill, J.T.: Mathematical programs with optimization problems in the constraints. Operations research **21**(1), 37–44 (1973)
6. Cournot, A.A.: Researches into the Mathematical Principles of the Theory of Wealth. New York: Macmillan Company, 1927 [c1897] (1927)
7. Daughety, A.F.: Beneficial concentration. The American Economic Review **80**(5), 1231–1237 (1990)
8. Debreu, G.: A social equilibrium existence theorem. Proceedings of the National Academy of Sciences **38**(10), 886–893 (1952)
9. Facchinei, F., Kanzow, C.: Generalized nash equilibrium problems. 4or **5**, 173–210 (2007)
10. Fukuda, K.: Cddlib reference manual. Report version 094a, Department of Mathematics, Institute of Theoretical Computer Science, ETH Zentrum, CH-8092 Zurich, Switzerland (2021)
11. Hu, M., Fukushima, M.: Multi-leader-follower games: models, methods and applications. Journal of the Operations Research Society of Japan **58**(1), 1–23 (2015)
12. Huck, S., Muller, W., Normann, H.T.: Stackelberg beats cournot—on collusion and efficiency in experimental markets. The Economic Journal **111**(474), 749–765 (2001)
13. Laine, F., Fridovich-Keil, D., Chiu, C.Y., Tomlin, C.: The computation of approximate generalized feedback nash equilibria. SIAM Journal on Optimization **33**(1), 294–318 (2023)
14. Nash, J.: Non-cooperative games. Annals of mathematics pp. 286–295 (1951)
15. Outrata, J.V.: On optimization problems with variational inequality constraints. SIAM Journal on optimization **4**(2), 340–357 (1994)
16. Rockafellar, R.T.: Convex analysis:(pms-28) (2015)
17. Sherali, H.D.: A multiple leader stackelberg model and analysis. Operations Research **32**(2), 390–404 (1984)
18. Stellato, B., Banjac, G., Goulart, P., Bemporad, A., Boyd, S.: OSQP: an operator splitting solver for quadratic programs. Mathematical Programming Computation **12**(4), 637–672 (2020). DOI 10.1007/s12532-020-00179-2. URL `https://doi.org/10.1007/s12532-020-00179-2`
19. Su, C.L.: Equilibrium problems with equilibrium constraints: Stationarities, algorithms, and applications. Stanford University (2005)
20. Ue-Pyng, W., Bialas, W.F.: The hybrid algorithm for solving the three-level linear programming problem. Computers & operations research **13**(4), 367–377 (1986)
21. Vicente, L.N., Calamai, P.H.: Bilevel and multilevel programming: A bibliography review. Journal of Global optimization **5**(3), 291–306 (1994)
22. Von Stackelberg, H.: Market structure and equilibrium. Springer Science & Business Media (2010)