A new branch-and-bound algorithm for standard quadratic programming problems

G. Liuzzi*, M. Locatelli† and Veronica Piccialli‡

OGDA December 21st 2018

* IASI-CNR Rome

† Dipartimento di Ingegneria e Architettura, Università di Parma

‡ Dipartimento di Ingegneria Civile e Ingegneria Informatica Università degli studi di Roma Tor Vergata

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Actually...

That is just the first part of the talk..

- Actually...
Introduction
Branch and Bound
Our bound
Bounding Strategy
Branching strategy
Numerical Results
Mixed Strategies Costs
The Algorithm
Results

Actually...

That is just the first part of the talk..

Second part of the talk:

A new branch and bound for finite Nash games with switching costs

Actually...
Introduction
Branch and Bound
Our bound
Bounding Strategy
Branching strategy
Numerical Results
Mixed Strategies Costs
The Algorithm
Results

Actually...

That is just the first part of the talk..

Second part of the talk:

A new branch and bound for finite Nash games with switching costs

Joint work with G. Liuzzi, M. Locatelli and Stefan Rass (System Security Group Universitat Klagenfurt)

- Actually...

Introduction
- StQP
- Complexity
- Our ingredients

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

StQP

The problem we consider is

$$\begin{aligned} \min \quad & \tfrac{1}{2} x^T Q x + c^T x \\ & x \in \Delta_n = \{x \in \mathbb{R}^n_+ \ : \ \textstyle\sum_{i=1}^n x_i = 1\}, \end{aligned} \tag{1}$$

where a quadratic function is minimized over the n-dimensional unit simplex

- Actually...

Introduction
- StQP
- Complexity
- Our ingredients

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

StQP

The problem we consider is

$$\begin{aligned} \min \quad & \tfrac{1}{2} x^T Q x + c^T x \\ & x \in \Delta_n = \{x \in \mathbb{R}^n_+ \ : \ \textstyle\sum_{i=1}^n x_i = 1\}, \end{aligned} \tag{1}$$

where a quadratic function is minimized over the n-dimensional unit simplex

Applications:

1. quadratic resource allocation problem
2. mean/variance portfolio selection problem
3. maximum clique
4. lower bound for the crossing number of complete bipartite graph $K_{m,n}$

Complexity

- Actually...

Introduction
- StQP
- Complexity
- Our ingredients

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

The problem is polinomially solvable when

- Actually...

Introduction
- StQP
- Complexity
- Our ingredients

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Complexity

The problem is polinomially solvable when

1. either $Q \succeq 0$ since the problem is convex

- Actually...

Introduction
- StQP
- Complexity
- Our ingredients

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Complexity

The problem is polinomially solvable when

1. either $Q \succeq 0$ since the problem is convex
2. or when the structure of Q is such that the optimum is attained at one of the n vertices of the simplex (i.e. $Q \preceq 0$ or $Q_{ii} = 0, Q \geq 0$)

- Actually...

Introduction
- StQP
- Complexity
- Our ingredients

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Complexity

The problem is polinomially solvable when

1. either $Q \succeq 0$ since the problem is convex
2. or when the structure of Q is such that the optimum is attained at one of the n vertices of the simplex (i.e. $Q \preceq 0$ or $Q_{ii} = 0, Q \geq 0$)

Otherwise it is NP-hard.

- Actually...

Introduction
- StQP
- Complexity
- Our ingredients

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Our ingredients

We propose a branch and bound approach that tries to implicitely enumerate all the KKT points of the problem. The algorithm exploits the following ideas:

1. an LP based bound based on the decomposition of the quadratic function into a sum of quadratic functions for which it is possible to compute the convex envelope over the unit simplex
2. a smart branching strategy
3. an improvement of the bound based on the KKT conditions

• Actually...
Introduction
Branch and Bound
• B&B
Our bound
Bounding Strategy
Branching strategy
Numerical Results
Mixed Strategies Costs
The Algorithm
Results

B&B

We denote by P_0 problem (1), i.e., the root node of the branch-and-bound tree, and L_0 its lower bound, by z the global UB.

$$\Pi = \{(P_0, L_0)\}, z = +\infty$$

While $\Pi \neq \emptyset$

$$(P, L) \leftarrow \texttt{select}(\Pi)$$
$$\Pi \leftarrow \Pi \setminus \{(P, L)\}$$
$$(\mathcal{C}, z) \leftarrow \texttt{branch\&bound}(P)$$
$$\texttt{update}(\Pi, \mathcal{C})$$

End

Return the global upper bound z

• Actually...
Introduction
Branch and Bound
• B&B
Our bound
Bounding Strategy
Branching strategy
Numerical Results
Mixed Strategies Costs
The Algorithm
Results

B&B

We denote by P_0 problem (1), i.e., the root node of the branch-and-bound tree, and L_0 its lower bound, by z the global UB.

$\Pi = \{(P_0, L_0)\}, z = +\infty$

While $\Pi \neq \emptyset$

$(P, L) \leftarrow \texttt{select}(\Pi)$
$\Pi \leftarrow \Pi \setminus \{(P, L)\}$
$(\mathcal{C}, z) \leftarrow \texttt{branch\&bound}(P)$
$\texttt{update}(\Pi, \mathcal{C})$

End

Return the global upper bound z

- the `select` procedure picks an open problem according to the selection strategy;

• Actually...
Introduction
Branch and Bound
• B&B
Our bound
Bounding Strategy
Branching strategy
Numerical Results
Mixed Strategies Costs
The Algorithm
Results

B&B

We denote by P_0 problem (1), i.e., the root node of the branch-and-bound tree, and L_0 its lower bound, by z the global UB.

$\Pi = \{(P_0, L_0)\}, z = +\infty$

While $\Pi \neq \emptyset$

$(P, L) \leftarrow \texttt{select}(\Pi)$
$\Pi \leftarrow \Pi \setminus \{(P, L)\}$
$(\mathcal{C}, z) \leftarrow \texttt{branch\&bound}(P)$
$\texttt{update}(\Pi, \mathcal{C})$

End

Return the global upper bound z

- the `select` procedure picks an open problem according to the selection strategy;
- the `branch&bound` , given an open problem P generates a set of children (branching strategy) and it computes the lower and upper bounds of the children, and possibly updates z;

- Actually...

Introduction

Branch and Bound

- B&B

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

B&B

We denote by P_0 problem (1), i.e., the root node of the branch-and-bound tree, and L_0 its lower bound, by z the global UB.

$\Pi = \{(P_0, L_0)\}, z = +\infty$

While $\Pi \neq \emptyset$

$(P, L) \leftarrow \texttt{select}(\Pi)$
$\Pi \leftarrow \Pi \setminus \{(P, L)\}$
$(\mathcal{C}, z) \leftarrow \texttt{branch\&bound}(P)$
$\texttt{update}(\Pi, \mathcal{C})$

End

Return the global upper bound z

- the `select` procedure picks an open problem according to the selection strategy;
- the `branch&bound` , given an open problem P generates a set of children (branching strategy) and it computes the lower and upper bounds of the children, and possibly updates z;
- the `update` procedure updates the set of currently open problems Π. Namely, it adds to Π the new problems in $\mathcal{C}$ and it (possibly) fathoms from Π those problems with a lower bound greater than or equal to the global upper bound.

- Actually...

Introduction

Branch and Bound

Our bound
- Observations
- Convexity graph
- Convex envelope

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Observations

Observation 1 *For any StQP problem (1):*

(i) *matrix Q and vector c can always be re-defined in such a way that $Q_{ii} = 0$ for all $i = 1, \ldots, n$, i.e., all diagonal elements of the Hessian matrix Q can be taken equal to 0;*

(ii) *if the diagonal condition above holds, then all $Q_{ij} > 0$ can be set equal to 0, since at any optimal solution of the StQP problem, $Q_{ij} > 0$ implies that at least one among x_i and x_j is certainly equal to 0.*

Indeed the objective can be rewritten as

$$\sum_{i,j} Q_{ij}x_ix_j + \sum_i c_ix_i + \sum_i Q_{ii}x_i - \sum_i Q_{ii}x_i(\sum_j x_j) = \sum_{i,j}(Q_{ij} - Q_{ii})x_ix_j + \sum_i (Q_{ii} + c_i)x_i$$

and it can be easily shown that if there exists a pair i, j such that $Q_{ij} > 0$ then either $x_i = 0$ or $x_j = 0$ in the optimal solution (by contradiction).

- Actually...

Introduction

Branch and Bound

Our bound
- Observations
- Convexity graph
- Convex envelope

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Convexity graph

A graph $G = (V, E)$, called **convexity graph**, can be associated to problem (1) [Scozzari Tardella 08], where

$$V = \{1, \ldots, n\}, \quad E = \{(i, j) \ : \ i, j \in \{1, \ldots, n\}, \ Q_{ij} < 0\}.$$

Observation 2 *The optimal solution of (1) must be attained in the relative interior of a face*

$$F_C = \{\mathbf{x} \in \Delta_n \ : \ x_i = 0, \ i \notin C\}, \tag{2}$$

where $C \subseteq V$ is a clique over the convexity graph G. This implies that, given any independent set $I \subseteq V$ in the convexity graph G, at most one x_i, $i \in I$, is strictly positive at the optimal solution of (1).

- Actually...

Introduction

Branch and Bound

Our bound

- Observations
- Convexity graph
- Convex envelope

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Convex envelope

Consider the quadratic function:

$$f^i(\mathbf{x}) = \sum_{j \neq i} A_{ij} x_i x_j,$$

whose convexity graph is a star with center the node corresponding to variable x_i

- Actually...

Introduction

Branch and Bound

Our bound

- Observations
- Convexity graph
- Convex envelope

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Convex envelope

Consider the quadratic function:

$$f^i(\mathbf{x}) = \sum_{j \neq i} A_{ij} x_i x_j,$$

whose convexity graph is a star with center the node corresponding to variable x_i

The convex envelope of f^i over the unit simplex can be computed analitically

- Actually...

Introduction

Branch and Bound

Our bound
- Observations
- Convexity graph
- Convex envelope

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Convex envelope

Consider the quadratic function:

$$f^i(\mathbf{x}) = \sum_{j \neq i} A_{ij} x_i x_j,$$

whose convexity graph is a star with center the node corresponding to variable x_i

The convex envelope of f^i over the unit simplex can be computed analitically

Starting from this convex envelope a weaker LP bound can be defined that is significantly cheaper without losing too much in quality :

$$\begin{aligned} \ell_{\mathrm{LP}}(Q, c) \;=\; \min \quad & \sum_{i=1}^{n} y_i + c^T \mathbf{x} \\ & \mathbf{x} \in \Delta_n \\ & y_i \geq \gamma_r^i(\mathbf{x}) \qquad i = 1, \ldots, n,\ r = 1, \ldots, t_i + 1. \end{aligned} \tag{3}$$

where γ_r^i is a supporting hyperplane of the convex envelope of f^i

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy
- KKT conditions
- Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

KKT conditions

The KKT conditions for problem (1) are

$$Q_i\mathbf{x} + c_i - \lambda = \mu_i \qquad i = 1, \ldots, n$$

$$\mu_i \geq 0 \qquad\qquad i = 1, \ldots, n,$$

where Q_i is the i-th row of matrix Q, λ is the Lagrange multiplier of the constraint $\sum_{i=1}^{n} x_i = 1$, and μ_i, $i = 1, \ldots, n$, is the Lagrange multiplier of the constraint $x_i \geq 0$.

Observation 3 *Let $(\mathbf{x}, \lambda, \boldsymbol{\mu})$ be a KKT point of (1). If $x_i > 0$, then for each $j \neq i$*

$$Q_j\mathbf{x} + c_j \geq Q_i\mathbf{x} + c_i. \tag{4}$$

Moreover

$$\frac{1}{2}\mathbf{x}^T Q\mathbf{x} + c^T\mathbf{x} \geq \frac{1}{2}\left[Q_i\mathbf{x} + c_i + c^T\mathbf{x}\right]. \tag{5}$$

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy
- KKT conditions
- Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Bounding Strategy

At each node of the B&B tree, we associate a vector $\mathbf{s} \in \{-1, 0, 1\}^n$ of variable statuses. Specifically,

$$s_i = \begin{cases} -1 & \text{implies } x_i \geq 0 \\ 0 & \text{implies } x_i = 0 \\ 1 & \text{implies } x_i > 0. \end{cases}$$

We compute a lower bound at a given node by solving an LP problem where we add the constraints deriving from the corresponding node status $\mathbf{s}$. More specifically,

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy
- KKT conditions
- Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Bounding Strategy

At each node of the B&B tree, we associate a vector $\mathbf{s} \in \{-1, 0, 1\}^n$ of variable statuses. Specifically,

$$s_i = \begin{cases} -1 & \text{implies } x_i \geq 0 \\ 0 & \text{implies } x_i = 0 \\ 1 & \text{implies } x_i > 0. \end{cases}$$

We compute a lower bound at a given node by solving an LP problem where we add the constraints deriving from the corresponding node status $\mathbf{s}$. More specifically,

i) for every i such that $s_i = 0$, the x_i variable is fixed to 0;

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy
- KKT conditions
- Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Bounding Strategy

At each node of the B&B tree, we associate a vector $\mathbf{s} \in \{-1, 0, 1\}^n$ of variable statuses. Specifically,

$$s_i = \begin{cases} -1 & \text{implies } x_i \geq 0 \\ 0 & \text{implies } x_i = 0 \\ 1 & \text{implies } x_i > 0. \end{cases}$$

We compute a lower bound at a given node by solving an LP problem where we add the constraints deriving from the corresponding node status $\mathbf{s}$. More specifically,

i) for every i such that $s_i = 0$, the x_i variable is fixed to 0;

ii) for every i such that $s_i = 1$, the constraints (4) are added; furthermore, exploiting the bound on the objective function (5) the following constraint is added

$$\sum_{j=1}^{n} y_j \geq \frac{1}{2} \left[Q_i \mathbf{x} + c_i + c^T \mathbf{x} \right];$$

finally, for every $j \neq i$ such that $Q_{ij} = 0$ and $s_j = -1$, variable x_j is fixed to 0; should s_j be equal to 1, the node can be fathomed since no optimal solution of Problem (1) can have both x_i and x_j greater than zero when $Q_{ij} = 0$ (see Observation 1).

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy
- KKT conditions
- Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Bounding Strategy

At each node of the B&B tree, we associate a vector $\mathbf{s} \in \{-1, 0, 1\}^n$ of variable statuses. Specifically,

$$s_i = \begin{cases} -1 & \text{implies } x_i \geq 0 \\ 0 & \text{implies } x_i = 0 \\ 1 & \text{implies } x_i > 0. \end{cases}$$

We compute a lower bound at a given node by solving an LP problem where we add the constraints deriving from the corresponding node status $\mathbf{s}$. More specifically,

i) for every i such that $s_i = 0$, the x_i variable is fixed to 0;

ii) for every i such that $s_i = 1$, the constraints (4) are added; furthermore, exploiting the bound on the objective function (5) the following constraint is added

$$\sum_{j=1}^{n} y_j \geq \frac{1}{2}\left[Q_i\mathbf{x} + c_i + c^T\mathbf{x}\right];$$

finally, for every $j \neq i$ such that $Q_{ij} = 0$ and $s_j = -1$, variable x_j is fixed to 0; should s_j be equal to 1, the node can be fathomed since no optimal solution of Problem (1) can have both x_i and x_j greater than zero when $Q_{ij} = 0$ (see Observation 1).

Note that whenever the matrix Q is sparse, fixing $s_i = 1$ implies fixing many other variables to zero, due to the high number of zero elements on row i, so that the size of the LP problem reduces substantially.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy
- Branching
- Independent Set
- Independent Set

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Branching

Given a generic node of the B&B tree, the branching procedure exploits Observation 2: given any independent set $I \subseteq V$ in the convexity graph G, at most one x_i, $i \in I$, is strictly positive at the optimal solution of (1).

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

- Branching
- Independent Set
- Independent Set

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Branching

Given a generic node of the B&B tree, the branching procedure exploits Observation 2: given any independent set $I \subseteq V$ in the convexity graph G, at most one x_i, $i \in I$, is strictly positive at the optimal solution of (1).

We exploit this observation by defining a (possibly) non-binary branching procedure. In particular, given an independent set I in G, we generate $|I| + 1$ children with statuses such that:

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

- Branching
- Independent Set
- Independent Set

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Branching

Given a generic node of the B&B tree, the branching procedure exploits Observation 2: given any independent set $I \subseteq V$ in the convexity graph G, at most one x_i, $i \in I$, is strictly positive at the optimal solution of (1).

We exploit this observation by defining a (possibly) non-binary branching procedure. In particular, given an independent set I in G, we generate $|I| + 1$ children with statuses such that:

- for every $k \in I$, a child is generated with $s_k = 1$ and $s_i = 0$ for all $i \in I$, $i \neq k$;

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

- Branching
- Independent Set
- Independent Set

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Branching

Given a generic node of the B&B tree, the branching procedure exploits Observation 2: given any independent set $I \subseteq V$ in the convexity graph G, at most one x_i, $i \in I$, is strictly positive at the optimal solution of (1).

We exploit this observation by defining a (possibly) non-binary branching procedure. In particular, given an independent set I in G, we generate $|I| + 1$ children with statuses such that:

- for every $k \in I$, a child is generated with $s_k = 1$ and $s_i = 0$ for all $i \in I$, $i \neq k$;
- a further child is generated with $s_k = 0$, for all $k \in I$.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

- Branching
- Independent Set
- Independent Set

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Branching

Given a generic node of the B&B tree, the branching procedure exploits Observation 2: given any independent set $I \subseteq V$ in the convexity graph G, at most one x_i, $i \in I$, is strictly positive at the optimal solution of (1).

We exploit this observation by defining a (possibly) non-binary branching procedure. In particular, given an independent set I in G, we generate $|I| + 1$ children with statuses such that:

- for every $k \in I$, a child is generated with $s_k = 1$ and $s_i = 0$ for all $i \in I$, $i \neq k$;
- a further child is generated with $s_k = 0$, for all $k \in I$.

Note that this branching strategy implies that, if the matrix is sparse, the computation of the first $|T|$ bounds should be fast, since fixing $s_i = 1$ implies fixing many other variables to zero, due to the high number of zero elements on row i, so that the size of the LP problem reduces substantially.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

- Branching
- Independent Set
- Independent Set

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Independent Set

Let $\bar{V} \subseteq V$ be the subset of nodes of graph G associated with variables whose statuses are equal to -1 at the current node, and $G[\bar{V}]$ the subgraph of G induced by the set $\bar{V}$. Moreover, let $(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ be the optimal solution of the relaxation at the current node.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

- Branching
- Independent Set
- Independent Set

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Independent Set

Let $\bar{V} \subseteq V$ be the subset of nodes of graph G associated with variables whose statuses are equal to -1 at the current node, and $G[\bar{V}]$ the subgraph of G induced by the set $\bar{V}$. Moreover, let $(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ be the optimal solution of the relaxation at the current node.

For every node $i \in \bar{V}$, let

$$\eta_i = \bar{y}_i - \bar{\mathbf{x}}^\top \mathbf{q}_i \bar{x}_i \leq 0, \text{ for each } i \in \bar{V},$$

and π be the permutation corresponding to the ascending ordering of η_i's.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

- Branching
- Independent Set
- Independent Set

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Independent Set

Let $\bar{V} \subseteq V$ be the subset of nodes of graph G associated with variables whose statuses are equal to -1 at the current node, and $G[\bar{V}]$ the subgraph of G induced by the set $\bar{V}$. Moreover, let $(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ be the optimal solution of the relaxation at the current node.

For every node $i \in \bar{V}$, let

$$\eta_i = \bar{y}_i - \bar{\mathbf{x}}^\top \mathbf{q}_i \bar{x}_i \leq 0, \text{ for each } i \in \bar{V},$$

and π be the permutation corresponding to the ascending ordering of η_i's.

We recall that the value η_i represents an estimate of the quality of the polyhedral bound on the star graph induced by node $i \in \bar{V}$. Hence, the lower this value, the more promising the node is to improve the bound.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

- Branching
- Independent Set
- Independent Set

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Independent Set

We compute an independent set I on $G[\bar{V}]$ by using a greedy algorithm proposed in [Blelloch et al. '12] using as ordering of the nodes in $\bar{V}$ the above defined π.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

- Branching
- Independent Set
- Independent Set

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

Independent Set

We compute an independent set I on $G[\bar{V}]$ by using a greedy algorithm proposed in [Blelloch et al. '12] using as ordering of the nodes in $\bar{V}$ the above defined π.

We further process the independent set I by defining the following subset of nodes $T \subset I$,

$$T = \left\{ i \in I : |\eta_i| \geq 1.5\, \bar{\eta}_I \right\}, \quad \text{with} \quad \bar{\eta}_I = \frac{1}{|I|} \sum_{i \in I} |\eta_i|.$$

This amounts to choosing only the more promising nodes, i.e., those having a gap η_i sufficiently away from zero. By using set T, we generate $|T| + 1$ children as described above.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Setting

We implemented our branch and bound algorithm using Julia v0.4.5 and ran our experiments on a 2.7 GHz Intel I5 PC, with 32GB of RAM.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Setting

We implemented our branch and bound algorithm using Julia v0.4.5 and ran our experiments on a 2.7 GHz Intel I5 PC, with 32GB of RAM.

GUROBI and Ipopt have been compiled with version 5.3.1 of the GCC library.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Setting

We implemented our branch and bound algorithm using Julia v0.4.5 and ran our experiments on a 2.7 GHz Intel I5 PC, with 32GB of RAM.

GUROBI and Ipopt have been compiled with version 5.3.1 of the GCC library.

Both the code and the instances on which it has been tested are freely available for download at the URL `http://www.iasi.cnr.it/~liuzzi/StQP`.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Setting

We implemented our branch and bound algorithm using Julia v0.4.5 and ran our experiments on a 2.7 GHz Intel I5 PC, with 32GB of RAM.

GUROBI and Ipopt have been compiled with version 5.3.1 of the GCC library.

Both the code and the instances on which it has been tested are freely available for download at the URL `http://www.iasi.cnr.it/~liuzzi/StQP`.

We compare with two approaches:

- [Scozzari Tardella 08] cheap $O(n^2)$ lower bounds are employed and an implicit enumeration is carried on over the set of cliques of the so called *convexity graph* of (1),
- [Bundfuss Duer 09], that is an adaptive linear approximation algorithm for copositive programs applied to the copositive formulation of StQP problems.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Random Instances

We employed the algorithm described in [Nowak 99] to randomly generate the test instances, used also in [Scozzari Tardella 08].

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Random Instances

We employed the algorithm described in [Nowak 99] to randomly generate the test instances, used also in [Scozzari Tardella 08].

In these instances it is possible to set a parameter, the density $d \in [0, 1]$ of the underlying convexity graph, through which the difficulty of the StQP problem can be varied. Namely, as d increases the problems become more difficult.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Random Instances

We employed the algorithm described in [Nowak 99] to randomly generate the test instances, used also in [Scozzari Tardella 08].

In these instances it is possible to set a parameter, the density $d \in [0, 1]$ of the underlying convexity graph, through which the difficulty of the StQP problem can be varied. Namely, as d increases the problems become more difficult.

For each couple (number of variables, density), we generated five random instances beside the one provided to us by Prof. A. Scozzari and also employed in [Scozzari Tardella 08].

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Random Instances

We employed the algorithm described in [Nowak 99] to randomly generate the test instances, used also in [Scozzari Tardella 08].

In these instances it is possible to set a parameter, the density $d \in [0, 1]$ of the underlying convexity graph, through which the difficulty of the StQP problem can be varied. Namely, as d increases the problems become more difficult.

For each couple (number of variables, density), we generated five random instances beside the one provided to us by Prof. A. Scozzari and also employed in [Scozzari Tardella 08].

We were able to solve efficiently problems of size $n \in \{100, 200, 500\}$ with $d \in \{0.25, 0.5, 0.75\}$, and the best strategy was the one with N-ary branching and best bound visiting strategy

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Results on random Instances

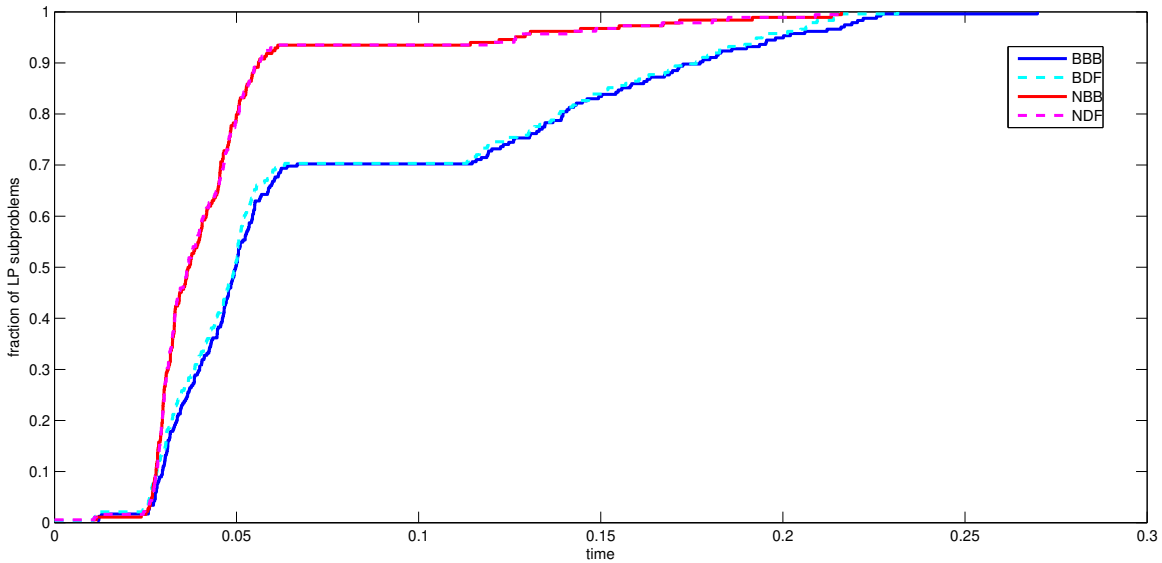

Figure 1: Percentage of LP problems solved within time t on a problem with $n = 200$ and $d = 0.25$

We report the solution time on the x-axis and the number of problems solved within that time for each strategy on the y-axis. As expected, the n-ary branching allows to solve more problems where the variables are fixed to be positive, resulting in faster LPs since many variables are then fixed to zero

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Comparison with [Scozzari Tardella 08]

n	0.25		0.5		0.75	
	NBB	[ST 08]	NBB	[ST 08]	NBB	[ST 08]
100	1.61	1.00	6.40	29.05	36.72	1021.03
200	6.40	8.47	66.79	502.71	1573.05	-

We report the computing times for NBB and for the approach proposed in [Scozzari Tardella 08], normalized by the time required by the latter to solve the problem with $n = 100$ and density 0.25 (which is, thus, equal to 1).

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Comparison with [Scozzari Tardella 08]

n	0.25		0.5		0.75	
	NBB	[ST 08]	NBB	[ST 08]	NBB	[ST 08]
100	1.61	1.00	6.40	29.05	36.72	1021.03
200	6.40	8.47	66.79	502.71	1573.05	-

We report the computing times for NBB and for the approach proposed in [Scozzari Tardella 08], normalized by the time required by the latter to solve the problem with $n = 100$ and density 0.25 (which is, thus, equal to 1).

The important observation that one can draw from these results is that, both at $n = 100$ and at $n = 200$, the increase of the computing times with the density of the underlying convexity graph is much slower for NBB so that that the proposed approach scales better than the approach in [Scozzari Tardella 08], as the density increases.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Comparison with [Scozzari Tardella 08]

n	0.25		0.5		0.75	
	NBB	[ST 08]	NBB	[ST 08]	NBB	[ST 08]
100	1.61	1.00	6.40	29.05	36.72	1021.03
200	6.40	8.47	66.79	502.71	1573.05	-

We report the computing times for NBB and for the approach proposed in [Scozzari Tardella 08], normalized by the time required by the latter to solve the problem with $n = 100$ and density 0.25 (which is, thus, equal to 1).

The important observation that one can draw from these results is that, both at $n = 100$ and at $n = 200$, the increase of the computing times with the density of the underlying convexity graph is much slower for NBB so that that the proposed approach scales better than the approach in [Scozzari Tardella 08], as the density increases.

Furthermore, we were able to solve exactly also instances they could only tackle with an heuristic approacH ($n = 500$)

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Comparison with [Bundfuss Duer 09]

In [Bundfuss Duer 09] they report results on random instances where the entries of matrix Q are uniformly random generated in the interval $[-n, n]$. For these instances excellent results are reported for dimension n up to 10,000.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Comparison with [Bundfuss Duer 09]

In [Bundfuss Duer 09] they report results on random instances where the entries of matrix Q are uniformly random generated in the interval $[-n, n]$. For these instances excellent results are reported for dimension n up to 10,000.

However, it turns out that these instances tend to be very simple, since , it turns out that for each arbitrary relative precision $\varepsilon > 0$, the vertex of the unit simplex with the lowest objective function value can be certified as an ε-optimal solution (by the trivial lower bound obtained by taking the minimum of all the entries of the matrix Q) with probability one as $n \to \infty$.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Comparison with [Bundfuss Duer 09]

In [Bundfuss Duer 09] they report results on random instances where the entries of matrix Q are uniformly random generated in the interval $[-n, n]$. For these instances excellent results are reported for dimension n up to 10,000.

However, it turns out that these instances tend to be very simple, since , it turns out that for each arbitrary relative precision $\varepsilon > 0$, the vertex of the unit simplex with the lowest objective function value can be certified as an ε-optimal solution (by the trivial lower bound obtained by taking the minimum of all the entries of the matrix Q) with probability one as $n \to \infty$.

We ran the algorithm proposed in [Bundfuss Duer 09] on our instances with time limit 3000 seconds:

n(density)	$(ub - lb)/\|ub\|$	iterations	cpu time
100(0.25) avg.	0.0000	720.5	185.14
100(0.5) avg.	0.0439	1647.833333	3002.64
100(0.75) avg.	0.1373	1692.833333	3004.80
200(0.25) avg.	0.1063	1496.5	3003.39
200(0.5) avg.	0.2158	1502.167	3002.66

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Comparison with [Bundfuss Duer 09]

In [Bundfuss Duer 09] they report results on random instances where the entries of matrix Q are uniformly random generated in the interval $[-n, n]$. For these instances excellent results are reported for dimension n up to 10,000.

However, it turns out that these instances tend to be very simple, since , it turns out that for each arbitrary relative precision $\varepsilon > 0$, the vertex of the unit simplex with the lowest objective function value can be certified as an ε-optimal solution (by the trivial lower bound obtained by taking the minimum of all the entries of the matrix Q) with probability one as $n \to \infty$.

We ran the algorithm proposed in [Bundfuss Duer 09] on our instances with time limit 3000 seconds:

n(density)	$(ub - lb)/\|ub\|$	iterations	cpu time
100(0.25) avg.	0.0000	720.5	185.14
100(0.5) avg.	0.0439	1647.833333	3002.64
100(0.75) avg.	0.1373	1692.833333	3004.80
200(0.25) avg.	0.1063	1496.5	3003.39
200(0.5) avg.	0.2158	1502.167	3002.66

We solved the instance `johnson8-2-4` in 14 seconds (and after the generation of 1,185 nodes) with respect to a computing time of 93 seconds reported in [Bundfuss Duer 09], while we solved the instance `hamming6-4` in 39 seconds (and 2,961 nodes) with respect to a computing time of almost 1 hour reported in [Bundfuss Duer 09].

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Hard instances

As a final experiment with our approach, we tested it over the instances recently proposed in [Bomze et al 2018].

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Hard instances

As a final experiment with our approach, we tested it over the instances recently proposed in [Bomze et al 2018].

They are built in such a way that the number of local minimizers grows exponentially with n and the difference between the function values of these local minimizers is very small.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Hard instances

As a final experiment with our approach, we tested it over the instances recently proposed in [Bomze et al 2018].

They are built in such a way that the number of local minimizers grows exponentially with n and the difference between the function values of these local minimizers is very small.

The density of their underlying convexity graph is $d = 1$. Such instances are obviously the worst possible ones for any method based on implicit enumeration, since the effect of pruning is very mild and almost a complete enumeration is needed.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

- Setting
- Random Instances
- Results on random Instances
- Comparison with [Scozzari Tardella 08]
- Comparison with [Bundfuss Duer 09]
- Hard instances

Mixed Strategies Costs

The Algorithm

Results

Hard instances

As a final experiment with our approach, we tested it over the instances recently proposed in [Bomze et al 2018].

They are built in such a way that the number of local minimizers grows exponentially with n and the difference between the function values of these local minimizers is very small.

The density of their underlying convexity graph is $d = 1$. Such instances are obviously the worst possible ones for any method based on implicit enumeration, since the effect of pruning is very mild and almost a complete enumeration is needed.

Matrix	BBB		BDF	
M24a	**357783**	6541.37	372839	6272.39
M24b	**324509**	5783.32	333647	4493.06
M24c	308013	5355.19	302773	4058.09

Table 1: Results on the matrices proposed in [Bomze et al 18] with $n = 24$

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

- Security Games
- An example
- Formulation of the problem
- How difficult is the problem?

The Algorithm

Results

Security Games

We are in the context of repeated games for security

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

- Security Games
- An example
- Formulation of the problem
- How difficult is the problem?

The Algorithm

Results

Security Games

We are in the context of repeated games for security

Playing a repeated game usually assumes that strategies can be changed without effort or costs between instances of the same game.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

- Security Games
- An example
- Formulation of the problem
- How difficult is the problem?

The Algorithm

Results

Security Games

We are in the context of repeated games for security

Playing a repeated game usually assumes that strategies can be changed without effort or costs between instances of the same game.

Suppose a game has an equilibrium in pure strategies. Then both players can straightforwardly implement their individually optimal action and the cost can be meaningfully subtracted from the revenue

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

- Security Games
- An example
- Formulation of the problem
- How difficult is the problem?

The Algorithm

Results

Security Games

We are in the context of repeated games for security

Playing a repeated game usually assumes that strategies can be changed without effort or costs between instances of the same game.

Suppose a game has an equilibrium in pure strategies. Then both players can straightforwardly implement their individually optimal action and the cost can be meaningfully subtracted from the revenue

However, what happens if the game has all its equilibria in mixed strategies? In that case, a player is forced to change its behavior over repetitions of the game, where the switch from one pure strategy played in the last round to the new (randomly chosen) strategy in the next round of the game is tied to some cost.

● Actually...
Introduction
Branch and Bound
Our bound
Bounding Strategy
Branching strategy
Numerical Results
Mixed Strategies Costs
● Security Games
● An example
● Formulation of the problem
● How difficult is the problem?
The Algorithm
Results

An example

Consider an enterprise building with rooms $R_1, \ldots, R_n$ to be visited by a security guard (player 1) repeatedly at random.

● Actually...
Introduction
Branch and Bound
Our bound
Bounding Strategy
Branching strategy
Numerical Results
Mixed Strategies Costs
● Security Games
● An example
● Formulation of the problem
● How difficult is the problem?
The Algorithm
Results

An example

Consider an enterprise building with rooms $R_1, \ldots, R_n$ to be visited by a security guard (player 1) repeatedly at random.

The visit to the i-th room corresponds to the i-th pure strategy in the action set of player 1

● Actually...
Introduction
Branch and Bound
Our bound
Bounding Strategy
Branching strategy
Numerical Results
Mixed Strategies Costs
● Security Games
● An example
● Formulation of the problem
● How difficult is the problem?
The Algorithm
Results

An example

Consider an enterprise building with rooms $R_1, \ldots, R_n$ to be visited by a security guard (player 1) repeatedly at random.

The visit to the i-th room corresponds to the i-th pure strategy in the action set of player 1

An equilibrium exists only in a mixed strategies (all the rooms need to be checked and more than once)

● Actually...
Introduction
Branch and Bound
Our bound
Bounding Strategy
Branching strategy
Numerical Results
Mixed Strategies Costs
● Security Games
● An example
● Formulation of the problem
● How difficult is the problem?
The Algorithm
Results

An example

Consider an enterprise building with rooms $R_1, \ldots, R_n$ to be visited by a security guard (player 1) repeatedly at random.

The visit to the i-th room corresponds to the i-th pure strategy in the action set of player 1

An equilibrium exists only in a mixed strategies (all the rooms need to be checked and more than once)

The guard would surely prefer checking nearby rooms at. once and leaving far away rooms for later

● Actually...
Introduction
Branch and Bound
Our bound
Bounding Strategy
Branching strategy
Numerical Results
Mixed Strategies Costs
● Security Games
● An example
● Formulation of the problem
● How difficult is the problem?
The Algorithm
Results

An example

Consider an enterprise building with rooms $R_1, \ldots, R_n$ to be visited by a security guard (player 1) repeatedly at random.

The visit to the i-th room corresponds to the i-th pure strategy in the action set of player 1

An equilibrium exists only in a mixed strategies (all the rooms need to be checked and more than once)

The guard would surely prefer checking nearby rooms at. once and leaving far away rooms for later

However, rooms may have different importance levels for the attacker (e.g., induced by different security clearances in the enterprise).

● Actually...
Introduction
Branch and Bound
Our bound
Bounding Strategy
Branching strategy
Numerical Results
Mixed Strategies Costs
● Security Games
● An example
● Formulation of the problem
● How difficult is the problem?
The Algorithm
Results

An example

Consider an enterprise building with rooms $R_1, \ldots, R_n$ to be visited by a security guard (player 1) repeatedly at random.

The visit to the i-th room corresponds to the i-th pure strategy in the action set of player 1

An equilibrium exists only in a mixed strategies (all the rooms need to be checked and more than once)

The guard would surely prefer checking nearby rooms at. once and leaving far away rooms for later

However, rooms may have different importance levels for the attacker (e.g., induced by different security clearances in the enterprise).

A shortest round trip route may not be optimal, since the security guard has to visit all the rooms with prescribed frequencies.

● Actually...
Introduction
Branch and Bound
Our bound
Bounding Strategy
Branching strategy
Numerical Results
Mixed Strategies Costs
● Security Games
● An example
● Formulation of the problem
● How difficult is the problem?
The Algorithm
Results

An example

Consider an enterprise building with rooms $R_1, \ldots, R_n$ to be visited by a security guard (player 1) repeatedly at random.

The visit to the i-th room corresponds to the i-th pure strategy in the action set of player 1

An equilibrium exists only in a mixed strategies (all the rooms need to be checked and more than once)

The guard would surely prefer checking nearby rooms at. once and leaving far away rooms for later

However, rooms may have different importance levels for the attacker (e.g., induced by different security clearances in the enterprise).

A shortest round trip route may not be optimal, since the security guard has to visit all the rooms with prescribed frequencies.

The game is repeated but the cost of playing strategy visit room R_i depends on which room has been checked just before. Thus, playing a mixed strategy equilibrium induces costs not when the strategy is played but mostly when the strategy is changed.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

- Security Games
- An example
- Formulation of the problem
- How difficult is the problem?

The Algorithm

Results

Formulation of the problem

Assume that player 1 has n strategies whereas player 2 has m strategies. Let S_{ij} be the random (according to a certain distribution $F_{S_{ij}}(x)$) cost for player 1 for switching from strategy i to strategy j

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

- Security Games
- An example
- Formulation of the problem
- How difficult is the problem?

The Algorithm

Results

Formulation of the problem

Assume that player 1 has n strategies whereas player 2 has m strategies. Let S_{ij} be the random (according to a certain distribution $F_{S_{ij}}(x)$) cost for player 1 for switching from strategy i to strategy j

Thanks to the low of total probability, the vector of switching cost for all strategies can be written as Sx, so that the switching cost in mixed strategies is given by $x^T Sx$.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

- Security Games
- An example
- Formulation of the problem
- How difficult is the problem?

The Algorithm

Results

Formulation of the problem

Assume that player 1 has n strategies whereas player 2 has m strategies. Let S_{ij} be the random (according to a certain distribution $F_{S_{ij}}(x)$) cost for player 1 for switching from strategy i to strategy j

Thanks to the low of total probability, the vector of switching cost for all strategies can be written as Sx, so that the switching cost in mixed strategies is given by $x^T Sx$.

Assuming that the player 1 gives priority $\alpha \in (0,1)$ to his payoff $u_1(x,y) = x^T Ay$, and priority $1-\alpha$ to the switching costs, so that his problem is

$$\begin{aligned} \min \quad & v \\ & v \geq \alpha x^T Sx + (1-\alpha) x^T Ae_i, \quad i = 1, \ldots, m \\ & e^T x = 1 \\ & x \geq 0 \end{aligned} \tag{6}$$

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

- Security Games
- An example
- Formulation of the problem
- How difficult is the problem?

The Algorithm

Results

How difficult is the problem?

The problem we want to solve is

$$\begin{aligned} \min \quad & \alpha x^T S x + v \\ & v \geq (1-\alpha) x^T A e_i, \quad i = 1, \ldots, m \\ & e^T x = 1 \\ & x \geq 0 \end{aligned} \tag{7}$$

where $S_{ii} = 0$ for $i = 1, \ldots, n$ and $S_{ij} \geq 0$ for $i, j = 1, \ldots, n, i < j$ (the matrix is indefinite) and $A \geq 0$

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

- Security Games
- An example
- Formulation of the problem
- How difficult is the problem?

The Algorithm

Results

How difficult is the problem?

The problem we want to solve is

$$\begin{aligned} \min \quad & \alpha x^T S x + v \\ & v \geq (1-\alpha) x^T A e_i, \quad i = 1, \ldots, m \\ & e^T x = 1 \\ & x \geq 0 \end{aligned} \tag{7}$$

where $S_{ii} = 0$ for $i = 1, \ldots, n$ and $S_{ij} \geq 0$ for $i, j = 1, \ldots, n, i < j$ (the matrix is indefinite) and $A \geq 0$

$\alpha = \mathbf{0}$ We have a linear problem (easy)

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

- Security Games
- An example
- Formulation of the problem
- How difficult is the problem?

The Algorithm

Results

How difficult is the problem?

The problem we want to solve is

$$\begin{aligned} \min \quad & \alpha x^T S x + v \\ & v \geq (1-\alpha) x^T A e_i, \quad i = 1, \ldots, m \\ & e^T x = 1 \\ & x \geq 0 \end{aligned} \tag{7}$$

where $S_{ii} = 0$ for $i = 1, \ldots, n$ and $S_{ij} \geq 0$ for $i, j = 1, \ldots, n, i < j$ (the matrix is indefinite) and $A \geq 0$

$\alpha = \mathbf{0}$ We have a linear problem (easy)

$\alpha = \mathbf{1}$ We have a pure StQP whose solution is any vertex of the simplex where the cost is 0.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

- Security Games
- An example
- Formulation of the problem
- How difficult is the problem?

The Algorithm

Results

How difficult is the problem?

The problem we want to solve is

$$\begin{aligned} \min \quad & \alpha x^T S x + v \\ & v \geq (1-\alpha) x^T A e_i, \quad i = 1, \ldots, m \\ & e^T x = 1 \\ & x \geq 0 \end{aligned} \tag{7}$$

where $S_{ii} = 0$ for $i = 1, \ldots, n$ and $S_{ij} \geq 0$ for $i, j = 1, \ldots, n, i < j$ (the matrix is indefinite) and $A \geq 0$

$\alpha = \mathbf{0}$ We have a linear problem (easy)

$\alpha = \mathbf{1}$ We have a pure StQP whose solution is any vertex of the simplex where the cost is 0.

$\alpha \in (\mathbf{0}, \mathbf{1})$ It can be proved that the problem is **NP-complete** since it is equivalent to the max clique decision problem.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm
- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Bounding Strategy

The objective of the problem is composed by a **quadratic term** and a **linear term**

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Bounding Strategy

The objective of the problem is composed by a **quadratic term** and a **linear term**

We cannot use anymore the bound based on the convex envelope since the convex envelope of $x^T Sx$ is the trivial zero bound

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Bounding Strategy

The objective of the problem is composed by a **quadratic term** and a **linear term**

We cannot use anymore the bound based on the convex envelope since the convex envelope of $x^T Sx$ is the trivial zero bound

We cannot use the KKT conditions as we did before

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Bounding Strategy

The objective of the problem is composed by a **quadratic term** and a **linear term**

We cannot use anymore the bound based on the convex envelope since the convex envelope of $x^T Sx$ is the trivial zero bound

We cannot use the KKT conditions as we did before

In this case the convexity graph is dense, so we cannot fix many x to zero

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Bounding Strategy

The objective of the problem is composed by a **quadratic term** and a **linear term**

We cannot use anymore the bound based on the convex envelope since the convex envelope of $x^T Sx$ is the trivial zero bound

We cannot use the KKT conditions as we did before

In this case the convexity graph is dense, so we cannot fix many x to zero

We can bound the linear part by using the actual Global Upper Bound

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm
- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Bounding Strategy

The objective of the problem is composed by a **quadratic term** and a **linear term**

We cannot use anymore the bound based on the convex envelope since the convex envelope of $x^T Sx$ is the trivial zero bound

We cannot use the KKT conditions as we did before

In this case the convexity graph is dense, so we cannot fix many x to zero

We can bound the linear part by using the actual Global Upper Bound

We bound the quadratic part by using Mc Cormick inequalities

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Lower Bound

To find a lower bound we solve the following problem:

$$
\begin{aligned}
(LPB) \quad & \min \quad t \\
& \text{s.t.} \quad t \geq v + \sum_{i=1}^{n} f_i \\
& v \geq x^T A e_i \quad i = 1, \ldots, m \\
& t \leq GUB \\
& v + \sum_{i=1}^{n} f_i \leq GUB \\
& f_i \geq 0.5 \left(min[l_i]x_i + min[i] \sum_{j=1}^{n} S_{ij}x_j - min[l_i]min[i] \right) \quad i = 1, \ldots, n \\
& f_i \geq 0.5 \left(max[l_i]x_i + max[i] \sum_{j=1}^{n} S_{ij}x_j - max[l_i]max[i] \right) \quad i = 1, \ldots, n \\
& e^T x = 1 \\
& x \geq 0
\end{aligned}
$$

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Lower Bound

To find a lower bound we solve the following problem:

$$
\begin{aligned}
(LPB) \quad & \min \quad t \\
& \text{s.t.} \quad t \geq v + \sum_{i=1}^{n} f_i \\
& v \geq x^T A e_i \quad i = 1, \ldots, m \\
& t \leq GUB \\
& v + \sum_{i=1}^{n} f_i \leq GUB \\
& f_i \geq 0.5 \left(min[l_i] x_i + min[i] \sum_{j=1}^{n} S_{ij} x_j - min[l_i] min[i] \right) \quad i = 1, \ldots, n \\
& f_i \geq 0.5 \left(max[l_i] x_i + max[i] \sum_{j=1}^{n} S_{ij} x_j - max[l_i] max[i] \right) \quad i = 1, \ldots, n \\
& e^T x = 1 \\
& x \geq 0
\end{aligned}
$$

1. $min[l_i]$ ($max[l_i]$) is the minimum (maximum) value at the node of $l_i(x) = \sum_{j=1}^{n} S_{ij} x_j$. At the root node is initialized at 0 ($\max_j \{S_{ij}\}$).

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Lower Bound

To find a lower bound we solve the following problem:

$$
\begin{aligned}
(LPB) \quad & \min \quad t \\
& \text{s.t.} \quad t \geq v + \sum_{i=1}^{n} f_i \\
& v \geq x^T A e_i \quad i = 1, \ldots, m \\
& t \leq GUB \\
& v + \sum_{i=1}^{n} f_i \leq GUB \\
& f_i \geq 0.5 \left(min[l_i] x_i + min[i] \sum_{j=1}^{n} S_{ij} x_j - min[l_i] min[i] \right) \quad i = 1, \ldots, n \\
& f_i \geq 0.5 \left(max[l_i] x_i + max[i] \sum_{j=1}^{n} S_{ij} x_j - max[l_i] max[i] \right) \quad i = 1, \ldots, n \\
& e^T x = 1 \\
& x \geq 0
\end{aligned}
$$

1. $min[l_i]$ ($max[l_i]$) is the minimum (maximum) value at the node of $l_i(x) = \sum_{j=1}^{n} S_{ij} x_j$. At the root node is initialized at 0 ($\max_j \{S_{ij}\}$).
2. $min[i]$ ($max[i]$) is the minimum (maximum) value at the node of x_i. It is initialized at 0 (1).

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Bounding Procedure

while there is improvement

1. Compute $min[l_i]$, $max[l_i]$, $max[i]$ by solving LPs (for all i or for an interesting subsets)
2. Solve problem (LPB)

end

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Bounding Procedure

while there is improvement

1. Compute $min[l_i]$, $max[l_i]$, $max[i]$ by solving LPs (for all i or for an interesting subsets)
2. Solve problem (LPB)

end

At the root node the bounding procedure is repeated until the improvement goes below a certain. tolerance (10^{-4}) whereas at the nodes there is a maximum number of times and the tolerance improvement is set to 10^{-2} to decrease the computational time

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Branching Strategy

We select for branching the index i that maximizes the gap between $l_i(x) = \sum_{j=1}^{n} S_{ij}x_j$ and its best McCormick limitation f_i at the node

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Branching Strategy

We select for branching the index i that maximizes the gap between $l_i(x) = \sum_{j=1}^{n} S_{ij}x_j$ and its best McCormick limitation f_i at the node

Let x_i^{LB} be the value of x_i at the solution of problem (LPB), and x_i^* the i-th component of the current global upper bound, and let $min[i]$ and $max[i]$ be the current bounds on variable x_i.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Branching Strategy

We select for branching the index i that maximizes the gap between $l_i(x) = \sum_{j=1}^{n} S_{ij}x_j$ and its best McCormick limitation f_i at the node

Let x_i^{LB} be the value of x_i at the solution of problem (LPB), and x_i^* the i-th component of the current global upper bound, and let $min[i]$ and $max[i]$ be the current bounds on variable x_i.

We have two cases:

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm
- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Branching Strategy

We select for branching the index i that maximizes the gap between $l_i(x) = \sum_{j=1}^{n} S_{ij}x_j$ and its best McCormick limitation f_i at the node

Let x_i^{LB} be the value of x_i at the solution of problem (LPB), and x_i^* the i-th component of the current global upper bound, and let $min[i]$ and $max[i]$ be the current bounds on variable x_i.

We have two cases:

(a) $x_i^* \in (min[i], max[i])$, then $\bar{x} = x^*$

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm
- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Branching Strategy

We select for branching the index i that maximizes the gap between $l_i(x) = \sum_{j=1}^{n} S_{ij}x_j$ and its best McCormick limitation f_i at the node

Let x_i^{LB} be the value of x_i at the solution of problem (LPB), and x_i^* the i-th component of the current global upper bound, and let $min[i]$ and $max[i]$ be the current bounds on variable x_i.

We have two cases:

(a) $x_i^* \in (min[i], max[i])$, then $\bar{x} = x^*$
(b) Otherwise $\bar{x} = x^{LB}$

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm
- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Branching Strategy

We select for branching the index i that maximizes the gap between $l_i(x) = \sum_{j=1}^{n} S_{ij}x_j$ and its best McCormick limitation f_i at the node

Let x_i^{LB} be the value of x_i at the solution of problem (LPB), and x_i^* the i-th component of the current global upper bound, and let $min[i]$ and $max[i]$ be the current bounds on variable x_i.

We have two cases:

(a) $x_i^* \in (min[i], max[i])$, then $\bar{x} = x^*$
(b) Otherwise $\bar{x} = x^{LB}$

Four children are generated:

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm
- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Branching Strategy

We select for branching the index i that maximizes the gap between $l_i(x) = \sum_{j=1}^{n} S_{ij}x_j$ and its best McCormick limitation f_i at the node

Let x_i^{LB} be the value of x_i at the solution of problem (LPB), and x_i^* the i-th component of the current global upper bound, and let $min[i]$ and $max[i]$ be the current bounds on variable x_i.

We have two cases:

(a) $x_i^* \in (min[i], max[i])$, then $\bar{x} = x^*$
(b) Otherwise $\bar{x} = x^{LB}$

Four children are generated:

1. $x_i \leq \bar{x}_i$, $S_{i,:}x \leq S_{i,:}\bar{x} \rightarrow$ update $max[i]$, $max[l_i]$ and the corresponding McCormick inequality is updated
2. $x_i \leq \bar{x}_i$, $S_{i,:}x \geq S_{i,:}\bar{x} \rightarrow$ update $max[i]$, $min[l_i]$ and the corresponding McCormick inequaliies are updated
3. $x_i \geq \bar{x}_i$, $S_{i,:}x \leq S_{i,:}\bar{x} \rightarrow$ update $min[i]$, $max[l_i]$ and the corresponding McCormick inequalities are updated
4. $x_i \geq \bar{x}_i$, $S_{i,:}x \geq S_{i,:}\bar{x} \rightarrow$ update $min[i]$, $min[l_i]$ and the corresponding McCormick inequality is updated

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Root Node

At the root node after the bounding procedure has been applied, there is also a spatial branching on the variable v (bounding the linear term), computing its maximum and minimum value $v_{\max}$ and $v_{\min}$, and dividing the interval in small sub intervals that are finer around v^* (value at the current GUB)

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

- Bounding Strategy
- Lower Bound
- Bounding Procedure
- Branching Strategy
- Root Node

Results

Root Node

At the root node after the bounding procedure has been applied, there is also a spatial branching on the variable v (bounding the linear term), computing its maximum and minimum value $v_{\max}$ and $v_{\min}$, and dividing the interval in small sub intervals that are finer around v^* (value at the current GUB)

To improve the speed of convergence, we substitute the constraints

$$v + \sum_{i=1}^{n} f_i \leq GUB, \quad t \leq GUB$$

with

$$v + \sum_{i=1}^{n} f_i \leq (1 - tol)GUB \quad t \leq (1 - tol)GUB$$

with $tol = 10^{-3}$

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

- Instances
- Switch costs
- Results
- Conclusions
- Thank you

Instances

The game is about spot checking a set of n places to guard them against an adversary. The places are spatially scattered, with a directed weighted graph describing the connections by an edge with a random length (and the graph is built strongly connected).

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

- Instances
- Switch costs
- Results
- Conclusions
- Thank you

Instances

The game is about spot checking a set of n places to guard them against an adversary. The places are spatially scattered, with a directed weighted graph describing the connections by an edge with a random length (and the graph is built strongly connected).

The payoffs in the game are given by Matrix A, and are interpreted as the loss that the defending player 1 suffers when checking place i while the attacker is at place j. So, the defender

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

- Instances
- Switch costs
- Results
- Conclusions
- Thank you

Instances

The game is about spot checking a set of n places to guard them against an adversary. The places are spatially scattered, with a directed weighted graph describing the connections by an edge with a random length (and the graph is built strongly connected).

The payoffs in the game are given by Matrix A, and are interpreted as the loss that the defending player 1 suffers when checking place i while the attacker is at place j. So, the defender

1. can miss the attacker ($i \neq j$), in which case there will be a Weibull-distributed random loss

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

- Instances
- Switch costs
- Results
- Conclusions
- Thank you

Instances

The game is about spot checking a set of n places to guard them against an adversary. The places are spatially scattered, with a directed weighted graph describing the connections by an edge with a random length (and the graph is built strongly connected).

The payoffs in the game are given by Matrix A, and are interpreted as the loss that the defending player 1 suffers when checking place i while the attacker is at place j. So, the defender

1. can miss the attacker ($i \neq j$), in which case there will be a Weibull-distributed random loss
2. can hit the attacker at $i = j$, in which case there is zero loss.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

- Instances
- Switch costs
- Results
- Conclusions
- Thank you

Instances

The game is about spot checking a set of n places to guard them against an adversary. The places are spatially scattered, with a directed weighted graph describing the connections by an edge with a random length (and the graph is built strongly connected).

The payoffs in the game are given by Matrix A, and are interpreted as the loss that the defending player 1 suffers when checking place i while the attacker is at place j. So, the defender

1. can miss the attacker ($i \neq j$), in which case there will be a Weibull-distributed random loss
2. can hit the attacker at $i = j$, in which case there is zero loss.

The defender is thus minimizing, and the attacker is maximizing. We consider the problem of the defender.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

- Instances
- Switch costs
- Results
- Conclusions
- Thank you

Instances

The game is about spot checking a set of n places to guard them against an adversary. The places are spatially scattered, with a directed weighted graph describing the connections by an edge with a random length (and the graph is built strongly connected).

The payoffs in the game are given by Matrix A, and are interpreted as the loss that the defending player 1 suffers when checking place i while the attacker is at place j. So, the defender

1. can miss the attacker ($i \neq j$), in which case there will be a Weibull-distributed random loss
2. can hit the attacker at $i = j$, in which case there is zero loss.

The defender is thus minimizing, and the attacker is maximizing. We consider the problem of the defender.

The Nash equilibrium then gives the optimal random choice of spot checks to minimize the average loss.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

- Instances
- Switch costs
- Results
- Conclusions
- Thank you

Instances

The game is about spot checking a set of n places to guard them against an adversary. The places are spatially scattered, with a directed weighted graph describing the connections by an edge with a random length (and the graph is built strongly connected).

The payoffs in the game are given by Matrix A, and are interpreted as the loss that the defending player 1 suffers when checking place i while the attacker is at place j. So, the defender

1. can miss the attacker ($i \neq j$), in which case there will be a Weibull-distributed random loss
2. can hit the attacker at $i = j$, in which case there is zero loss.

The defender is thus minimizing, and the attacker is maximizing. We consider the problem of the defender.

The Nash equilibrium then gives the optimal random choice of spot checks to minimize the average loss.

To avoid trivialities, the payoff matrices were constructed not to admit pure strategy equilibria, so that the optimum (without switching cost) is necessarily a mixed strategy.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

- Instances
- Switch costs
- Results
- Conclusions
- Thank you

Switch costs

If the defender is currently at position i and next needs to check the (non-adjacent) place j, then the cost for the switch from i to j is the shortest path in the aforementioned graph . Since the graph is directed, the matrix is generally nonsymmetric.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

- Instances
- Switch costs
- Results
- Conclusions
- Thank you

Switch costs

If the defender is currently at position i and next needs to check the (non-adjacent) place j, then the cost for the switch from i to j is the shortest path in the aforementioned graph . Since the graph is directed, the matrix is generally nonsymmetric.

The weights in the graph are chosen exponentially distributed with rate parameter $\lambda = 0.2$, and the Weibull distribution for the losses has a shape parameter 5 and scale parameter about 10.63, so that both distributions have the same variance of 5.

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

- Instances
- Switch costs
- Results
- Conclusions
- Thank you

Switch costs

If the defender is currently at position i and next needs to check the (non-adjacent) place j, then the cost for the switch from i to j is the shortest path in the aforementioned graph . Since the graph is directed, the matrix is generally nonsymmetric.

The weights in the graph are chosen exponentially distributed with rate parameter $\lambda = 0.2$, and the Weibull distribution for the losses has a shape parameter 5 and scale parameter about 10.63, so that both distributions have the same variance of 5.

The size of the instances ranges from $n = 50$ to $n = 200$, but for the security application the most interesting ones are the ones with n around 50.

Results

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

- Instances
- Switch costs
- Results
- Conclusions
- Thank you

n	α	$GAP < 1\%$		$GAP \leq 0.1\%$	
		# nodes	time	# nodes	time
50	0.25	1	3.331	1	3.331
50	0.35	10	127.70	130	194.473
50	0.5	35	60.276	170	125.36
50	0.65	195	113.99	498	177.14
50	0.75	339	155.71	614	184.7
50	0.85	10	6.94	10	6.94
75	0.25	1	9.16	1	9.16
75	0.35	10	75.96	2054	1205.27
75	0.5	10	148.49	31290	27378.15
75	0.65	215	318.88	4022	2564.95
75	0.75	2271	2412.97	8082	4787.73
75	0.85	463	314.9	662	342.29
100	0.25	1	11.63	1	11.63
100	0.35	10	252.63	10	252.63
100	0.5	10	572.86		
100	0.75	3307	7686.14	99478	118842.09
100	0.85	855	1320.47	1350	1470.33

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

- Instances
- Switch costs
- Results
- Conclusions
- Thank you

Conclusions

We propose a B&B approach tailored for solving finite games with switching costs, that is a NP complete problem

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

- Instances
- Switch costs
- Results
- Conclusions
- Thank you

Conclusions

We propose a B&B approach tailored for solving finite games with switching costs, that is a NP complete problem

The algorithm is extremely effective in reducing the gap below 1%, while it is slow in closing the gap at dimension $n \geq 80$

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

- Instances
- Switch costs
- Results
- Conclusions
- Thank you

Conclusions

We propose a B&B approach tailored for solving finite games with switching costs, that is a NP complete problem

The algorithm is extremely effective in reducing the gap below 1%, while it is slow in closing the gap at dimension $n \geq 80$

The instances are easy for $\alpha \leq 0.35$ and $\alpha \geq 0.8$ whereas they are much harder for values of α close to 0.5

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

- Instances
- Switch costs
- Results
- Conclusions
- Thank you

Conclusions

We propose a B&B approach tailored for solving finite games with switching costs, that is a NP complete problem

The algorithm is extremely effective in reducing the gap below 1%, while it is slow in closing the gap at dimension $n \geq 80$

The instances are easy for $\alpha \leq 0.35$ and $\alpha \geq 0.8$ whereas they are much harder for values of α close to 0.5

From a practical point of view the interesting instances have a size close to $n = 50$ and a gap of 1% would already be enough

Thank you

- Actually...

Introduction

Branch and Bound

Our bound

Bounding Strategy

Branching strategy

Numerical Results

Mixed Strategies Costs

The Algorithm

Results

- Instances
- Switch costs
- Results
- Conclusions
- Thank you

Happy birthday Manuel!