Wiretapping: the nucleolus of connectivity

Haris Aziz
Oded Lachish
Mike Paterson
Rahul Savani

Department of Computer Science
University of Warwick
Spanning connectivity game

- Graph $G = (V, E)$
- Players $E$
- Coalitions $S \subseteq E$

$v(S) = \begin{cases} 
1 & \text{if } S \text{ induces a connected spanning subgraph} \\
0 & \text{otherwise}
\end{cases}$
Players rewarded for contributing to the connectivity of the graph

Show that the nucleolus can be computed efficiently

Relate connectivity game to non-cooperative wiretap game
Payoffs and excesses

- Payoff $x = (x_1, \ldots, x_{|E|})$ is a probability distribution
Payoffs and excesses

- Payoff $x = (x_1, \ldots, x_{|E|})$ is a probability distribution.
- For $S \subseteq E$, the excess is $e(x, S) = (\sum_{i \in S} x_i) - v(S)$. 

\[ x = (x_1, x_2, x_3, x_4) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{6}, \frac{1}{6}) \]
Payoffs and excesses

- Payoff \( x = (x_1, \ldots, x_{|E|}) \) is a probability distribution.
- For \( S \subseteq E \), the excess is \( e(x, S) = (\sum_{i \in S} x_i) - v(S) \).

\( (x_1, x_2, x_3, x_4) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{6}, \frac{1}{6}) \)

<table>
<thead>
<tr>
<th>S</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>23</th>
<th>24</th>
<th>34</th>
<th>123</th>
<th>124</th>
<th>134</th>
<th>234</th>
</tr>
</thead>
<tbody>
<tr>
<td>v(S)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>e(x, S)</td>
<td>( \frac{1}{3} )</td>
<td>( \frac{1}{3} )</td>
<td>( \frac{1}{6} )</td>
<td>( \frac{1}{6} )</td>
<td>( -\frac{1}{3} )</td>
<td>( -\frac{1}{2} )</td>
<td>( -\frac{1}{2} )</td>
<td>( -\frac{1}{2} )</td>
<td>( \frac{1}{3} )</td>
<td>( -\frac{1}{6} )</td>
<td>( -\frac{1}{6} )</td>
<td>( -\frac{1}{3} )</td>
<td>( -\frac{1}{3} )</td>
<td></td>
</tr>
</tbody>
</table>
The least core

Payoff \( x \) is in the \( \epsilon \)-core if

\[
e(x, S) \geq -\epsilon \quad \text{for all } S \subseteq E
\]

and the least core if it is in the \( \epsilon \)-core for smallest possible \( \epsilon \)
The least core

- Payoff $x$ is in the $\epsilon$-core if
  \[ e(x, S) \geq -\epsilon \quad \text{for all} \quad S \subseteq E \]

  and the least core if it is in the $\epsilon$-core for smallest possible $\epsilon$

- The least core is the set of solutions to this LP:

  \[
  \begin{align*}
  \text{minimize} & \quad \epsilon \\
  \text{subject to} & \quad e(x, S) \geq -\epsilon \quad \text{for all} \quad S \subseteq E \\
  & \quad x \geq 0 \\
  & \quad \sum_{i \in E} x_i = 1
  \end{align*}
  \]
The nucleolus

- The excess vector of \( x \) is

\[
(e(x, S_1), \ldots, e(x, S_{2^n}))
\]

where

\[
e(x, S_1) \leq e(x, S_2) \leq \cdots \leq e(x, S_{2^n})
\]
The nucleolus

- The **excess vector** of \( x \) is

\[(e(x, S_1), \ldots, e(x, S_{2^n}))\]

where

\[e(x, S_1) \leq e(x, S_2) \leq \cdots \leq e(x, S_{2^n})\]

- The **nucleolus** has the largest excess vector lexicographically
The nucleolus

- The **excess vector** of \( x \) is

\[
(e(x, S_1), \ldots, e(x, S_{2^n}))
\]

where

\[
e(x, S_1) \leq e(x, S_2) \leq \cdots \leq e(x, S_{2^n})
\]

- The **nucleolus** has the largest excess vector lexicographically

- Denote the distinct excesses by

\[
(-\epsilon_1, \ldots, -\epsilon_t) \text{ with } \epsilon_1 > \cdots > \epsilon_t
\]
The nucleolus

- The excess vector of $x$ is

$$ (e(x, S_1), \ldots, e(x, S_{2^n})) $$

where

$$ e(x, S_1) \leq e(x, S_2) \leq \cdots \leq e(x, S_{2^n}) $$

- The nucleolus has the largest excess vector lexicographically

- Denote the distinct excesses by

$$ (-\epsilon_1, \ldots, -\epsilon_t) $$ with $\epsilon_1 > \cdots > \epsilon_t$

- We call coalitions that obtain $-\epsilon_i$ as excess, $\epsilon_i$-coalitions
Example

\[(x_1, x_2, x_3, x_4) = \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{6}, \frac{1}{6}\right)\]

<table>
<thead>
<tr>
<th></th>
<th>S</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>23</th>
<th>24</th>
<th>34</th>
<th>123</th>
<th>124</th>
<th>134</th>
<th>234</th>
</tr>
</thead>
<tbody>
<tr>
<td>v(S)</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>e(x, S)</td>
<td></td>
<td>(\frac{1}{3})</td>
<td>(\frac{1}{3})</td>
<td>(\frac{1}{6})</td>
<td>(\frac{1}{6})</td>
<td>-(\frac{1}{3})</td>
<td>-(\frac{1}{2})</td>
<td>-(\frac{1}{2})</td>
<td>-(\frac{1}{2})</td>
<td>-(\frac{1}{2})</td>
<td>-(\frac{1}{2})</td>
<td>-(\frac{1}{2})</td>
<td>-(\frac{1}{2})</td>
<td>-(\frac{1}{6})</td>
<td>-(\frac{1}{6})</td>
</tr>
</tbody>
</table>
Goal: characterize nucleolus and provide efficient algorithm to find it
Proof strategy

**Goal:** characterize nucleolus and provide efficient algorithm to find it

1. Graph decomposition
Proof strategy

**Goal:** characterize nucleolus and provide efficient algorithm to find it

1. Graph decomposition
2. Decomposition of member of least core
Goal: characterize nucleolus and provide efficient algorithm to find it

1. Graph decomposition
2. Decomposition of member of least core
3. Relate these two decompositions
Goal: characterize nucleolus and provide efficient algorithm to find it

1. Graph decomposition
2. Decomposition of member of least core
3. Relate these two decompositions
4. Finding the nucleolus
Definition

For $G = (V, E)$ and $E' \subseteq E$, we define $C_G(E')$ as the set of connected components in the graph $G' = (V, E \setminus E')$.
For $G = (V, E)$ and $E' \subseteq E$, we define $C_G(E')$ as the set of connected components in the graph $G' = (V, E \setminus E')$. 
For $G = (V, E)$ and $E' \subseteq E$, we define $C_G(E')$ as the set of connected components in the graph $G' = (V, E \setminus E')$.
Definition

For $G = (V, E)$ and $E' \subseteq E$, we define $C_G(E')$ as the set of connected components in the graph $G' = (V, E \setminus E')$. 
Definition

\[
\text{opt}_G = \max_{E_1 \subseteq E, \ E_1 \neq \emptyset} \frac{|C_G(E_1)| - |C_G(\emptyset)|}{|E_1|}
\]
Definition

$$\text{opt}_G = \max_{E_1 \subseteq E, \ E_1 \neq \emptyset} \frac{|C_G(E_1)| - |C_G(\emptyset)|}{|E_1|}$$
\[
\text{opt}_G = \max_{E_1 \subseteq E, \ E_1 \neq \emptyset} \frac{|C_G(E_1)| - |C_G(\emptyset)|}{|E_1|}
\]
Definition

$$\text{opt}_G = \max_{E_1 \subseteq E, \ E_1 \neq \emptyset} \frac{|C_G(E_1)| - |C_G(\emptyset)|}{|E_1|}$$
A graph decomposition is a sequence \((G_1, E_1), \ldots, (G_t, E_t)\) so that for \(k = 1, \ldots, t\) we have \(E_k \subseteq E(G_k)\) and \(G_k = G \setminus \bigcup_{i=1,\ldots,k-1} E_i\).

It is optimal if for all \(k = 1, \ldots, t\), we have \(\text{opt}_{G_k} = \text{opt}_G\) and minimal optimal if \(E_k\) is minimal optimal for \(G_k\).
A graph decomposition is a sequence \((G_1, E_1), \ldots, (G_t, E_t)\) so that for \(k = 1, \ldots, t\) we have \(E_k \subseteq E(G_k)\) and \(G_k = G \setminus \bigcup_{i=1,\ldots,k-1} E_i\).

It is optimal if for all \(k = 1, \ldots, t\), we have \(\text{opt}_{G_k} = \text{opt}_G\) and minimal optimal if \(E_k\) is minimal optimal for \(G_k\).
Definition

A graph decomposition is a sequence \((G_1, E_1), \ldots, (G_t, E_t)\) so that for \(k = 1, \ldots, t\) we have \(E_k \subseteq E(G_k)\) and \(G_k = G \setminus \bigcup_{i=1, \ldots, k-1} E_i\).

It is optimal if for all \(k = 1, \ldots, t\), we have \(\text{opt}_{G_k} = \text{opt}_G\) and minimal optimal if \(E_k\) is minimal optimal for \(G_k\).
Definition

A graph decomposition is a sequence \((G_1, E_1), \ldots, (G_t, E_t)\) so that for \(k = 1, \ldots, t\) we have \(E_k \subseteq E(G_k)\) and \(G_k = G \setminus \bigcup_{i=1,\ldots,k-1} E_i\).

It is optimal if for all \(k = 1, \ldots, t\), we have \(\text{opt}_{G_k} = \text{opt}_G\) and minimal optimal if \(E_k\) is minimal optimal for \(G_k\).
**Definition**

A graph decomposition is a sequence $(G_1, E_1), \ldots, (G_t, E_t)$ so that for $k = 1, \ldots, t$ we have $E_k \subseteq E(G_k)$ and $G_k = G \setminus \bigcup_{i=1,\ldots,k-1} E_i$.

It is **optimal** if for all $k = 1, \ldots, t$, we have $\text{opt}_{G_k} = \text{opt}_G$ and **minimal optimal** if $E_k$ is minimal optimal for $G_k$. 

![Graph decomposition diagram]
Definition

A graph decomposition is a sequence $(G_1, E_1), \ldots, (G_t, E_t)$ so that for $k = 1, \ldots, t$ we have $E_k \subseteq E(G_k)$ and $G_k = G \setminus \bigcup_{i=1,\ldots,k-1} E_i$.

It is optimal if for all $k = 1, \ldots, t$, we have $\text{opt}_{G_k} = \text{opt}_G$ and minimal optimal if $E_k$ is minimal optimal for $G_k$. 
Definition

A graph decomposition is a sequence \((G_1, E_1), \ldots, (G_t, E_t)\) so that for \(k = 1, \ldots, t\) we have \(E_k \subseteq E(G_k)\) and \(G_k = G \setminus \bigcup_{i=1,\ldots,k-1} E_i\).

It is optimal if for all \(k = 1, \ldots, t\), we have \(\text{opt}_{G_k} = \text{opt}_G\) and minimal optimal if \(E_k\) is minimal optimal for \(G_k\).
Let $y_1 > \ldots > y_m > 0$ be the distinct payoffs of $x$.
Decomposition sequence

Definition

- Let $y_1 > \ldots > y_m > 0$ be the distinct payoffs of $x$
- Let $E_i \subseteq E$ be the edges with weight $y_i$ for $i = 1, \ldots, m$
Decomposition sequence

Definition

- Let $y_1 > \ldots > y_m > 0$ be the distinct payoffs of $x$
- Let $E_i \subseteq E$ be the edges with weight $y_i$ for $i = 1, \ldots, m$
- We call $E_1, \ldots, E_m$ the decomposition sequence of $x$
Decomposition sequence

Definition

- Let $y_1 > \ldots > y_m > 0$ be the distinct payoffs of $x$
- Let $E_i \subseteq E$ be the edges with weight $y_i$ for $i = 1, \ldots, m$
- We call $E_1, \ldots, E_m$ the decomposition sequence of $x$

For the decomposition sequence of a member of the least core there is some optimal graph decomposition that refines it
Member of the least core

- Consider $G^x = (V, E, x)$ for $x = (x_1, \ldots, x_{|E|})$ in least core
Member of the least core

- Consider $G^x = (V, E, x)$ for $x = (x_1, \ldots, x_{|E|})$ in least core
- Let $w$ be the weight of an MST of $G^x$
Consider $G^x = (V, E, x)$ for $x = (x_1, \ldots, x_{|E|})$ in least core.

Let $w$ be the weight of an MST of $G^x$.

All MSTs of $G^x$ are $\epsilon_1$-coalitions for $x$.

$\epsilon_1 = \epsilon$ is equal to $1 - w$.
Participation lemma

Every edge $e \in E$ features in at least one MST of $G^x$
Every edge $e \in E$ features in at least one MST of $G^x$. So the decomposition sequence of any member of the least core induces a graph decomposition. We show that this decomposition is optimal.
Participation lemma

Every edge $e \in E$ features in at least one MST of $G^x$
Participation lemma

Every edge $e \in E$ features in at least one MST of $G^x$.
Participation lemma

Every edge $e \in E$ features in at least one MST of $G^x$
Participation lemma

Every edge $e \in E$ features in at least one MST of $G^x$

So the decomposition sequence of any member of the least core induces a graph decomposition.
Participation lemma

Every edge $e \in E$ features in at least one MST of $G^\times$

So the decomposition sequence of any member of the least core induces a graph decomposition

We show that this decomposition is optimal
Decomposition sequence and MSTs

\[ |E(T) \cap E_1| = |C_G(E_1)| - 1 \]
|E(T) ∩ E_1| = |C_G(E_1)| − 1
Pushing weights

In any MST for this member of the least core, we will use one red edge and one yellow edge. Red edges have strictly larger weight than yellow edges.
In any MST for this member of the least core, we will use one red edge and one yellow edge.
In any MST for this member of the least core, we will use one red edge and one yellow edge.

Red edges have strictly larger weight than yellow edges.
Choose $\delta$ strictly smaller than half the difference between weights of reds and yellows. Shift $\delta$ from each red edge to the yellow edges, so the weight of any MST would increase by $\delta$. Move weight from bad ratio to better ratio to get contradiction (except where bad ratio has zero weight). All have same ratio.
Choose $\delta$ strictly smaller than half the difference between weights of reds and yellows.
Choose $\delta$ strictly smaller than half the difference between weights of reds and yellows

Shift $\delta$ from each red edge to the yellow edges, so the weight of any MST would increase by $\delta$
Choose $\delta$ strictly smaller than half the difference between weights of reds and yellows

Shift $\delta$ from each red edge to the yellow edges, so the weight of any MST would increase by $\delta$

Move weight from bad ratio to better ratio to get contradiction (except where bad ratio has zero weight)
Choose $\delta$ strictly smaller than half the difference between weights of reds and yellows.

Shift $\delta$ from each red edge to the yellow edges, so the weight of any MST would increase by $\delta$.

Move weight from bad ratio to better ratio to get contradiction (except where bad ratio has zero weight).

All have same ratio.
The union of all of them has the same ratio.

If we average all of them in the union, we still have a member of the least core.

(moving weight between two sets with the same ratio doesn't change the weight of an MST)

If the ratio was not optimal, we could divide the weight equally over the edge set with the ratio of the optimal graph to get a better MST.

We have an optimal graph decomposition.
The union of all of them has same ratio

If we average all in union we still have a member of least core (moving weight between two sets with same ratio doesn't change weight of an MST).

If the ratio was not $\text{opt}_G$, we could divide weight equally over edge set with ratio $\text{opt}_G$ to get a better MST.

We have an optimal graph decomposition.

- The union of all of them has same ratio
The union of all of them has same ratio

If we average all in union we still have a member of least core (moving weight between two sets with same ratio doesn’t change weight of an MST)
The union of all of them has same ratio

If we average all in union we still have a member of least core (moving weight between two sets with same ratio doesn’t change weight of an MST)

If the ratio was not $\text{opt}_G$, we could divide weight equally over edge set with ratio $\text{opt}_G$ to get a better MST
The union of all of them has same ratio

If we average all in union we still have a member of least core (moving weight between two sets with same ratio doesn’t change weight of an MST)

If the ratio was not $\text{opt}_G$, we could divide weight equally over edge set with ratio $\text{opt}_G$ to get a better MST

We have an optimal graph decomposition
For the decomposition sequence of a member of the least core there is some optimal graph decomposition that refines it
Corollary

Let \((G_1, E_1), \ldots, (G_t, E_t)\) be a minimal optimal graph decomposition. For every element of the least core

- All edges in \(E_i\) get the same payoff for \(i = 1, \ldots, t\)
Corollary

Let \((G_1, E_1), \ldots, (G_t, E_t)\) be a minimal optimal graph decomposition. For every element of the least core

- All edges in \(E_i\) get the same payoff for \(i = 1, \ldots, t\)
- All edges in \(E(G) \setminus \bigcup_{i=1,\ldots,t} E_i\) get payoff 0
Supports of the least core

**Corollary**

Let \((G_1, E_1), \ldots, (G_t, E_t)\) be a minimal optimal graph decomposition. For every element of the least core

- All edges in \(E_i\) get the same payoff for \(i = 1, \ldots, t\)
- All edges in \(E(G) \setminus \bigcup_{i=1,\ldots,t} E_i\) get payoff 0

**Definition** *(c-imputation \(\alpha = (\alpha_1, \ldots, \alpha_t)\))*

For \(i = 1, \ldots, t\), every edge \(e \in E_i\) gets payoff \(\alpha_i \geq 0\).
(G_i, E_i) parent of (G_j, E_j) for i < j if we can replace an e ∈ E_j in some least core MST with an e′ ∈ E_i and get a spanning tree
Layered graph decomposition DAG

- \((G_i, E_i)\) parent of \((G_j, E_j)\) for \(i < j\) if we can replace an \(e \in E_j\) in some least core MST with an \(e' \in E_i\) and get a spanning tree
Layered graph decomposition DAG

- \((G_i, E_i)\) parent of \((G_j, E_j)\) for \(i < j\) if we can replace an \(e \in E_j\) in some least core MST with an \(e' \in E_i\) and get a spanning tree
Claim

If \((G_i, E_i)\) is a parent of \((G_j, E_j)\), we have \(\alpha_i \geq \alpha_j\) for all \(c\)-imputations \(\alpha\) in the least core, and

\[ \alpha_i > \alpha_j > 0 \]

for the \(c\)-imputation \(\beta\) of the nucleolus

- For nucleolus: minimize number of \(\epsilon_1\)-coalitions
Inequalities for least core and nucleolus

Claim

If \((G_i, E_i)\) is a parent of \((G_j, E_j)\), we have \(\alpha_i \geq \alpha_j\) for all \(c\)-imputations \(\alpha\) in the least core, and

\[
\alpha_i > \alpha_j > 0
\]

for the \(c\)-imputation \(\beta\) of the nucleolus

- For nucleolus: minimize number of \(\epsilon_1\)-coalitions
- Achieved by strict inequality between layers, restricting the MSTs
Inequalities for least core and nucleolus

**Claim**

If \((G_i, E_i)\) is a parent of \((G_j, E_j)\), we have \(\alpha_i \geq \alpha_j\) for all \(c\)-imputations \(\alpha\) in the least core, and

\[\alpha_i > \alpha_j > 0\]

for the \(c\)-imputation \(\beta\) of the nucleolus

- For nucleolus: minimize number of \(\epsilon_1\)-coalitions
- Achieved by strict inequality between layers, restricting the MSTs
- Making weights within a layer equal maximizes

\[\epsilon_1 - \epsilon_2 = w_2 - w_1\]
Upper bounds on $\epsilon_1 - \epsilon_2$

- Weight of any edge $e$ in $L_i$ (just add $e$ to an $\epsilon_1$-coalition)
Upper bounds on $\epsilon_1 - \epsilon_2$

- Weight of any edge $e$ in $L_i$ (just add $e$ to an $\epsilon_1$-coalition)
- By replacability, the difference between the weight of the edges of a parent and child in the layered graph decomposition
Guess a solution

- Every edge in $L_i$ gets weight $i \times \kappa$ for fixed $\kappa$

Proof
Guess a solution

- Every edge in $L_i$ gets weight $i \times \kappa$ for fixed $\kappa$
- Any other solution would give a worse difference $\epsilon_1 - \epsilon_2$

Proof
Guess a solution

- Every edge in $L_i$ gets weight $i \times \kappa$ for fixed $\kappa$
- Any other solution would give a worse difference $\epsilon_1 - \epsilon_2$

Proof

- $\epsilon_2 = \epsilon_1 - \kappa$ by definition
Guess a solution

- Every edge in $L_i$ gets weight $i \times \kappa$ for fixed $\kappa$
- Any other solution would give a worse difference $\epsilon_1 - \epsilon_2$

Proof

- $\epsilon_2 = \epsilon_1 - \kappa$ by definition
- For any other choice of payoff either
Guess a solution

- Every edge in $L_i$ gets weight $i \times \kappa$ for fixed $\kappa$
- Any other solution would give a worse difference $\epsilon_1 - \epsilon_2$

Proof

- $\epsilon_2 = \epsilon_1 - \kappa$ by definition
- For any other choice of payoff either
  1. Difference between weight of a parent and child is more than $\kappa$
Guess a solution

- Every edge in $L_i$ gets weight $i \times \kappa$ for fixed $\kappa$
- Any other solution would give a worse difference $\epsilon_1 - \epsilon_2$

Proof

- $\epsilon_2 = \epsilon_1 - \kappa$ by definition
- For any other choice of payoff either
  1. Difference between weight of a parent and child is more than $\kappa$
  2. An edge in $L_1$ has weight more than $\kappa$
Guess a solution

- Every edge in \( L_i \) gets weight \( i \times \kappa \) for fixed \( \kappa \)
- Any other solution would give a worse difference \( \epsilon_1 - \epsilon_2 \)

Proof

- \( \epsilon_2 = \epsilon_1 - \kappa \) by definition
- For any other choice of payoff either
  1. Difference between weight of a parent and child is more than \( \kappa \)
  2. An edge in \( L_1 \) has weight more than \( \kappa \)
- But then there is a difference between parent and child or a weight in \( L_1 \) that is less than \( \kappa \)
\[ \kappa = \frac{1}{3 \times (2) + 2 \times (2) + 1 \times (6 + 6)} = \frac{1}{22} \]
\[ \kappa = \frac{1}{3 \times (2) + 2 \times (2) + 1 \times (6 + 6)} = \frac{1}{22} \]

\[ \beta = (\beta_1, \beta_2, \beta_3, \beta_4) = \left( \frac{3}{22}, \frac{2}{22}, \frac{1}{22}, \frac{1}{22} \right) \]
Wiretapping

The least core is the set of solutions to this linear program:

\[ \text{minimize} \quad \epsilon \]

subject to

\[ e(x, S) \geq -\epsilon \quad \text{for all } S \in S \]

\[ x \geq 0 \]

\[ \sum_{i \in E} x_i = 1 \]
Wiretapping

- The least core is the set of solutions to this linear program:

  minimize \( 1 - z \)

  subject to \( e(x, S) \geq -(1 - z) \) for all \( S \in S \)

  \( x \geq 0 \)

  \( \sum_{i \in E} x_i = 1 \)

- These are the maxmin strategies of the wiretapper
The least core is the set of solutions to this linear program:

\[
\text{minimize} \quad 1 - z \\
\text{subject to} \quad e(x, S) \geq -(1 - z) \quad \text{for all } S \in S \\
x \geq 0 \\
\sum_{i \in E} x_i = 1
\]

These are the maxmin strategies of the wiretapper

Nucleolus minimizes number of best responses of opponent and maximizes penalty for non-optimal play of opponent