Subgraph and Supergraph Problems in \( r \)-tournaments

Narayanaswamy N S
Kanthi Kiran S

January 5, 2011
Motivation

- Directed feedback vertex problem is fixed parameter tractable in general directed graphs but only tournaments have known $O^*(c^k)$ algorithms ($c$ is a constant and $k$ is the maximum solution size allowed).

We study a class of graphs, named r-tournaments, which naturally bridges the gap between tournaments and general graphs.
Motivation

- Directed feedback vertex problem is fixed parameter tractable in general directed graphs but only tournaments have known $O^*(c^k)$ algorithms ($c$ is a constant and $k$ is the maximum solution size allowed).
- We study a class of graphs, named $r$-tournaments, which naturally bridges the gap between tournaments and general graphs.
A directed graph is called $r$-tournament, if every pair of vertices has a directed path of length $\leq r \in \mathbb{N}$ connecting them.
A directed graph is called an $r$-tournament, if every pair of vertices has a directed path of length $\leq r \in \mathbb{N}$ connecting them.

Clearly by this definition, a 1-tournament is a tournament and a connected directed graph on $n$ vertices is an $n$-tournament.
Feedback vertex set and c-dominating set in r-tournaments
Feedback vertex set

Theorem

An algorithm to test if a 2-tournament has a FVS of size atmost \( k \) in \( O^*(c^k) \) time can be used to test if a directed graph has a FVS of size atmost \( k \) in \( O^*(c^k) \) for some constant \( c \in \mathbb{R} \).

Thus the feedback vertex set has, in the parameterized sense, equivalent complexity in general directed graphs as tournaments.
Construction

- Given a graph $G$ on $n$ vertices, we encode each vertex using $\log_2 n$ bits.
Construction

- Given a graph $G$ on $n$ vertices, we encode each vertex using $\log_2 n$ bits.
- We add three groups of vertices: $\{u_1, u_2, \ldots, u_{\log_2 n}\}$, $\{w_1, w_2, \ldots, w_{\log_2 n}\}$, $\{z_1, z_2, \ldots, z_{\log_2 n}\}$.

For every element $u_i$, we add an edge from $u_i$ to a vertex $v$ of $G$ if the $i$-th element of the latter's binary representation is 0. Otherwise, we add an edge from $v$ to $u_i$. Remaining connections are as shown in the following example.
Construction

- Given a graph \( G \) on \( n \) vertices, we encode each vertex using \( \log_2 n \) bits.
- We add three groups of vertices: \( \{u_1, u_2, \ldots, u_{\log_2 n}\} \), \( \{w_1, w_2, \ldots, w_{\log_2 n}\} \), \( \{z_1, z_2, \ldots, z_{\log_2 n}\} \).
- For every element \( u_i \), we add an edge from \( u_i \) a vertex \( v \) of \( G \) if the \( i^{th} \) element of the latter’s binary representation is 0. Otherwise we add an edge from \( v \) to \( u_i \). Remaining connections are as shown in following example.
Example of a graph on 8 vertices

* Thick arrows imply every vertex of the color group is adjacent to the other color group preserving the direction.
Example of a graph on 8 vertices

* Thick arrows imply every vertex of the color group is adjacent to the other color group preserving the direction.
* Thick arrows imply every vertex of the color group is adjacent to the other color group preserving the direction.
* Thick arrows imply every vertex of the color group is adjacent to the other color group preserving the direction.
Example of a graph on 8 vertices

* Thick arrows imply every vertex of the color group is adjacent to the other color group preserving the direction.
Example of a graph on 8 vertices

* Thick arrows imply every vertex of the color group is adjacent to the other color group preserving the direction.
Example of a graph on 8 vertices

* Thick arrows imply every vertex of the color group is adjacent to the other color group preserving the direction.
It is clear that the constructed graph is a 2-tournament.
It is clear that the constructed graph is a 2-tournament.

If $k$ is the number of vertices to be deleted from $G$ to destroy all directed cycles from it, to make the constructed graph acyclic we must delete exactly $k + \log_2 n$.
Proof Outline

- It is clear that the constructed graph is a 2-tournament.

- If $k$ is the number of vertices to be deleted from $G$ to destroy all directed cycles from it, to make the constructed graph acyclic we must delete exactly $k + \log_2 n$.

- The key observation is that at least one of the color groups black, orange, yellow must be completely destroyed.
Also by deleting all yellow vertices along with the $k$ vertices of $G$ we can completely destroy cycles in the constructed graph.
Also by deleting all yellow vertices along with the $k$ vertices of $G$ we can completely destroy cycles in the constructed graph.

To test if a given graph has a directed FVS of size $k$, we convert the graph into the above 2-tournament.
Also by deleting all yellow vertices along with the $k$ vertices of $G$ we can completely destroy cycles in the constructed graph.

To test if a given graph has a directed FVS of size $k$, we convert the graph into the above 2-tournament.

If there is an algorithm to solve for FVS of size $k$ in 2-tournament, running in time $O^*(c^k)$ the said procedure will yield an algorithm to test FVS of size $k$ in general digraphs, with running time $O^*(c^{k+\log_2 n}) = O^*(c^k)$.
2c-dominating set in c-tournaments

**Theorem (Landau)**

*The set containing the vertex of maximum outdegree in a tournament forms a minimum 2-dominating set.*
The set containing the vertex of maximum outdegree in a tournament forms a minimum 2-dominating set.

Let $T$ be a $c$-tournament and $v$ a vertex with a maximum number of vertices at a directed distance at most $c$. Then $\{v\}$ is a $(2c)$-dominating set of $T$. 
Proof

Let $N_c(v)$ denote the set of vertices at directed distance atmost $c$.

Let $u$ be a vertex such that $v \not\in N_2^c(v)$.

The above assumption along with the condition that $T$ is a $c$-tournament implies that $N_c(v) \cup \{v\} \subseteq N_c(u)$.

This is impossible as $v$ has maximum cardinality of $N_c(v)$.
Let $N_c(v)$ denote the set of vertices at directed distance atmost $c$. Let $u \in T$ be a vertex such that $v \notin N_{2c}(v)$.
Proof

- Let $N_c(v)$ denote the set of vertices at directed distance atmost $c$.
  Let $u \in T$ be a vertex such that $v \notin N_{2c}(v)$.
- The above assumption along with the condition that $T$ is $c$-tournament implies that $N_c(v) \cup \{v\} \subseteq N_c(u)$. 

Proof

- Let $N_c(v)$ denote the set of vertices at directed distance atmost $c$. Let $u \in T$ be a vertex such that $v \notin N_{2c}(v)$.
- The above assumption along with the condition that $T$ is $c$-tournament implies that $N_c(v) \cup \{v\} \subseteq N_c(u)$.
- This is impossible as $v$ has maximum cardinality of $N_c(v)$.
Theorem

The $c$-dominating set problem restricted to $c$-tournaments is $W[2]$ complete under standard parameterization.
The c-dominating set problem restricted to c-tournaments is \( \mathbf{W}[2] \) complete under standard parameterization.

Also:

\[
\log_2 n
\]
The c-dominating set problem restricted to c-tournaments is \textit{W}\lbrack 2\rbrack complete under standard parameterization.

Also:

- Every c-tournament has a c-dominating set of size \(\log_2 n\). This results in an (brute force) algorithm to find out a c-dominating set, running in \(O(n^{\log_2 n})\).
Outline of the reduction

Given a tournament $T$, we replace each vertex of $v \in T$ by a path of $c$ vertices $v_i \ni i \in [c]$. If $(u, v) \in T$, we add edges from $u_i \ni i \in [c]$ to $v_1$.

The resultant graph $C_T$ is a $c$-tournament and has a $c$-dominating set of size $k_T$ has a dominating set of size $k$.
Given a tournament $T$, we replace each vertex of $v \in T$ by a path of $c$ vertices $v_i \ni i \in [c]$. If $(u, v) \in T$, we add edges from $u_i \ni i \in [c]$ to $v_1$. The resultant graph $CT$ is a $c$-tournament and has a $c$-dominating set of size $k$.
Outline of the reduction

- Given a tournament $T$, we replace each vertex of $v \in T$ by a path of $c$ vertices $v_i \ni i \in [c]$. If $(u, v) \in T$, we add edges from $u_i \ni i \in [c]$ to $v_1$.
- The resultant graph $CT$ is a $c$-tournament and has a $c$-dominating set of size $k$ iff $T$ has a dominating set of size $k$. 
If $D$ is a dominating set of $T$ then
\[
\{u_1 \ni u \in D\}
\]
is a c-dominating set of $CT$, with the same cardinality.

If $CD$ is a c-dominating set of $CT$, then the set of vertices in $D$ obtained by removing subscripts from elements of $CD$ is a dominating set of size at most $|CD|$. To see that $D$ is indeed the dominating set of $T$, observe that if $D$ does not dominate a vertex $v \in T$, then $CD$ does not dominate $v_c$. 

Naarayanawamy N S Kanthi Kiran S

Subgraph and Supergraph Problems in r-tournaments

January 5, 2011

13 / 27
If $D$ is a dominating set of $T$ then $\{u_1 \ni u \in D\}$ is a $c$-dominating set of $CT$, with the same cardinality.
If $D$ is a dominating set of $T$ then $\{u_1 \ni u \in D\}$ is a c-dominating set of $CT$, with the same cardinality.

If $CD$ is a c-dominating set of $CT$, then the set of vertices in $D$ obtained by removing subscripts from elements of $CD$ is a dominating set of size at most $|CD|$.
If $D$ is a dominating set of $T$ then $\{u_1 \ni u \in D\}$ is a $c$-dominating set of $CT$, with the same cardinality.

If $CD$ is a $c$-dominating set of $CT$, then the set of of vertices in $D$ obtained by removing subscripts from elements of $CD$ is a dominating set of size at most $|CD|$.

To see that $D$ is indeed the dominating set of $T$, observe that if $D$ does not dominate a vertex $v \in T$ $CD$ does not dominate $v_c$. 

Example: 3-dominating set in 3-tournament
Example: 3-dominating set in 3-tournament
Graph Modification Problems
A graph modification problem asks for an optimum number of modifications to a graph to obtain another one which satisfies some required property (a property is a class of graphs closed under isomorphism), example: the cluster editing problem.

We study two problems requiring modifications (edge addition and deletions) to obtain 2-tournaments (a cluster of 2-tournaments in the first problem and a single 2-tournament in the second case).
Problem (2-tournament clustering by edge deletion)

Given a digraph $G$, remove atmost $k$ edges to convert into a cluster of 2-tournaments.
Problem (2-tournament clustering by edge deletion)

*Given a digraph $G$, remove atmost $k$ edges to convert into a cluster of 2-tournaments.*

Problem (2-tournament completion)

*Given a digraph $G$, add atmost $k$ edges to convert into a 2-tournament.*
A Couple of Graph Modification Problems

Problem (2-tournament clustering by edge deletion)

Given a digraph $G$, remove atmost $k$ edges to convert into a cluster of 2-tournaments.

Problem (2-tournament completion)

Given a digraph $G$, add atmost $k$ edges to convert into a 2-tournament.

We prove that both 2-tournament clustering and 2-tournament completion are NP-Complete. We also prove that while 2-tournament clustering is FPT, 2-tournament completion is $W[2]$-hard.
2-tournament clustering is NPC: Reduction from clique clustering problem

Given a graph $G = (V, E)$, we construct $G' = (V', E')$ as following:

$$V' = \{u_+, u_- : u \in V\}.$$  \hspace{1cm} (1)

$$E' = \{(v_+, v_-) : v \in V\} \cup \{(v_-, u_+), (u_-, v_+)\}.$$  \hspace{1cm} (2)
2-tournament clustering is NPC: Reduction from clique clustering problem

Given a graph $G = (V, E)$, we construct $G' = (V', E')$ as following:

$$V' = \{u_+, u_- : u \in V\}.$$  \hfill (1)

$$E' = \{(v_+, v_-) : v \in V\} \cup \{(v_-, u_+), (u_-, v_+)\}.$$ \hfill (2)
The following is the series of steps involved in proving the NP-Hardness:

- Any subgraph of $G'$ which is a 2-tournament must have atmost one +ve signed vertex without a pair and atmost one -ve vertex without a pair.
- There is a minimum solution $M$ for the problem of 2-tournament edge clustering such that every vertex of $G'$ has its pair in one component.
- For each $k \geq 0$, an undirected graph $G$ has a clique clustering edge set of size atmost $k$ if and only if $G'$ has a 2-tournament clustering edge set of size $2k$. 
2-tournament clustering is FPT

Lemma

Let $G$ be a directed graph which is not a 2-tournament such that the underlying directed graph is connected. There exist two vertices for which the distance in the undirected graph is at most 3 but are not at directed distance 2 in $G$.

Proof.

1. Let $S$ be the set of pairs of vertices not having a directed path of length 2 connecting them. Let $u,v$ be a pair of vertices having least undirected distance among all pairs of $S$. Let the shortest undirected path connecting them be $P(u,v) = \{u,v_1,v_2,v_3..v\}$.

2. Let if possible $|P(u,v)| >= 4$. This means that $v_3 \neq v$ and $(u,v_3)$ does not belong to $S$. Hence there is a 2-path connecting $u,v_3$ which would imply $P$ is not the shortest path.
A simple search tree algorithm is based on the following observation: If the given graph is not a cluster of 2-tournament the earlier lemma gives us a pair of vertices which are not in the at a directed distance 2 but are at an undirected distance at most 3.

These vertices cannot be in the same component of the solution graph. Hence at least one of the edges on the path connecting \( u, v \) must be included in the final solution. Branching on each of these solutions yields a \( O^*(3^k) \) algorithm.
We prove the NP Completeness and W[2] hardness of the following problem:

**Problem (Single Vertex Satisfaction)**

Given a directed graph $G = (V, E)$ and a vertex $v \in V$ add at most $k$ edges to $G$ such that in the resultant graph every vertex of $G$ is either at directed distance at most 2 from $v$ or has it at a directed distance at most 2.
SVS is NPC and W[2] hard

- Reduction from dominating set problem which is NPC and W[2]C.
- Let $G = A \cup B$ (partitions $A$, $B$) be a bipartite graph. We add a new vertex $\nu$ to $G$ and direct the edges from $A$ to $B$ to get $G'$, as shown.
SVS is NPC and \( W[2] \) hard

- Reduction from dominating set problem which is NPC and \( W[2] \)C.
- Let \( G = A \cup B \) (partitions \( A, B \)) be a bipartite graph. We add a new vertex \( v \) to \( G \) and direct the edges from \( A \) to \( B \) to get \( G' \), as shown.
Proof outline

If $D$ is a dominating set of $G$, by adding edges edges from $v$ to all vertices of $D \cap A$ and from all vertices of $D \cap B$ to $v$ we get a graph in which $v$ satisfies all the 2-tournament property with all vertices.
Proof outline

- If \( D \) is a dominating set of \( G \), by adding edges from \( v \) to all vertices of \( D \cap A \) and from all vertices of \( D \cap B \) to \( v \) we get a graph in which \( v \) satisfies all the 2-tournament property with all vertices.
Let $M$ be a (edge set) solution to the SVS instance $(G', v)$. We prove that there is a dominating set of size $|M|$.

Let $M = M_G \cup M_v$, where $M_G$ edges whose end points are in $G$ and $M_v$ has edges incident on $v$.

Let $D_{G'}$ be the minimum dominating set of (underlying undirected graph) $G'$ and $D_G$ be the minimum dominating set of $G$. 
Adding $k$ edges to graph $G$ reduces the dominating set size by $k$ atmost:

$$D_{G'} \geq DG - |M_G|$$  \hspace{1cm} (3)

Since $v$ is at distance 2 from all the vertices in the underlying undirected graph and $M_v$ is its neighborhood, the latter is a dominating set of $G'$.

$$D_{G'} \leq |M_v|$$  \hspace{1cm} (4)

From the above equations we have:

$$|M_G| + |M_v| = k \geq D(G)$$  \hspace{1cm} (5)
2-tournament completion is NPC and \( \text{W}[2] \) hard: Reduction from Single Vertex Satisfaction

Given a graph \( G = (V, E) \) we construct \( G' = (V', E') \) in the following way. \( G' \) has an SVS edge set of size \( k \) iff \( G \) has a 2-tournament completion edge set of size \( k \):

\[
V' = V \cup V_1 \cup \{v_{ex}\}
\]

\[
V_1 = \{v_{u,w} : \forall \{u,w\} \in V - \{v\}\} \tag{6}
\]

\[
E' = E \cup \{(u, v_{u,w}), (v_{u,w}, w), \forall v_{u,w} \in V_1\}
\]

\[
\cup \{(u, v_{ex}), \forall u \in V\} \cup \{(v_{ex}, v_{u,w}), \forall v_{u,w} \in V_1\}
\]

\[
\cup \{(u, v) \forall \{u,v\} \in V_1\} \tag{7}
\]