Extremal Out-branchings and Out-trees in Digraphs

Gregory Gutin

Department of Computer Science
Royal Holloway, University of London

February 2008
Outline

1. Introduction
2. Classic Complexity
3. Fixed-Parameter Complexity
Outline

1. Introduction
2. Classic Complexity
3. Fixed-Parameter Complexity
We say that a subdigraph $T$ of a digraph $D$ is an **out-tree** if $T$ is an oriented tree with only one vertex $s$ of in-degree zero (its **root**).

The vertices of $T$ of out-degree zero are **leaves**.

If $T$ is a spanning out-tree, i.e. $V(T) = V(D)$, then $T$ is an **out-branching** of $D$.

A digraph $D$ has an out-branching iff $D$ has only one initial strong component (a strong component $C$ is **initial** if there are no arcs entering $C$).
Out-Branching Example
Outline

1 Introduction

2 Classic Complexity

3 Fixed-Parameter Complexity
MinLeafOBranch: Find an out-branching with minimum number of leaves (NP-hard)

MinLeafOBranch on acyclic digraphs: polynomial-time solvable

MaxLeafOBranch: Find an out-branching with maximum number of leaves (NP-hard even for undirected graphs)

MaxLeafOTree: Find an out-branching with maximum number of leaves (NP-hard)
MinLeafOBBranch on Acyclic Digraphs

Algorithm: $D \rightarrow B(D) \rightarrow M \rightarrow M^* \rightarrow T$
Out-Trees vs Out-Branchings-1

- $\ell_T(D)$ max no. leaves in an out-tree
- $\ell_B(D)$ max no. leaves in an out-branching
- $\ell_B(D) = \ell_T(D)$ or 0 for many digraphs including strong digraphs, acyclic digraphs, semicomplete multipartite digraphs, quasi-transitive digraphs, etc. (family $\mathcal{L}$ of digraphs); all undirected graphs are in $\mathcal{L}$. 
There are many digraphs outside $\mathcal{L}$:
$\ell_B(D) = 1$, $\ell_T(D) = n - 2$. $D \notin \mathcal{L}$. 
Outline

1. Introduction
2. Classic Complexity
3. Fixed-Parameter Complexity
**Definition**
A parameterized problem $\Pi$ can be considered as a set of pairs $(I, k)$ where $I$ is the problem instance and $k$ (usually an integer) is the parameter.

**Definition**
$\Pi$ is called fixed-parameter tractable (FPT) if membership of $(I, k)$ in $\Pi$ can be decided in time $O(f(k)|I|^c)$, where $|I|$ is the size of $I$, $f(k)$ is a computable function, and $c$ is a constant independent from $k$ and $I$. 
**Parameterized MaxLeafOT (M. Fellows)**

<table>
<thead>
<tr>
<th>Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Is it FPT to check whether there is an out-tree with at least k leaves?</em></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Theorem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Let $D \in \mathcal{L}$. Then either $\ell_B(D) \geq k$ or the underlying graph of $D$ is of pathwidth $\leq 2k^2$.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fact</th>
</tr>
</thead>
</table>
| Let $R_v$ be the set of vertices reachable from $v$. Then $D[R_v] \in \mathcal{L}$. Also, $\ell_T(D) = \max \{ \ell_B(D[R_v]) : v \in V(D) \}$.

<table>
<thead>
<tr>
<th>Theorem (Alon, Fomin, G., Krivelevich, Saurabh, ICALP’07)</th>
</tr>
</thead>
<tbody>
<tr>
<td>The problem is FPT.</td>
</tr>
</tbody>
</table>

1-Opt OB [Alon, Fomin, G., Krivelevich, Saurabh, FSTTCS’07]
Parameterized MaxLeafOB (M. Fellows)

**Problem**

*Is it FPT to check whether there is an out-branching with at least k leaves?*

**Theorem (Bonsma and Dorn, 2007)**

*The problem is FPT.*

Similar to Alon et al. but:

- Delete *useless* arcs from $D$
- Using an 1-Opt out-branching and some of its backward arcs, try to construct an out-branching with at least $k$ leaves. If no success, then pathwidth($UG(D)$) $\leq 6k^3$. 
Parameterized Problems for MinLeafOBBranch [G., Kim and Razgon, 2008]

- Check whether there is an out-branching with at most \( k \) leaves. It is NP-complete for each fixed integer \( k > 0 \).
- Check whether there is an out-branching with at most \( n/k \) leaves. It is NP-complete for each fixed integer \( k > 0 \).
- Check whether there is an out-branching with at most \( n - k \) leaves (i.e., at least \( k \) non-leaves). It is FPT.
Parameterized MinLeafOBranch: Definitions

Let $T$ be an out-branching. The parent $x$ of a leaf is of type 1 (2) if $d^+(x) = 1$ ($d^+(x) > 1$). A leaf is of type $i$ if its parent is of type $i$. An out-branching $T$ is normalized if there is no arc $uv$ in $D$ such that $u$ and $v$ are leaves and $v$ is of type 2.

**Fact**

*The set of leaves of type 2 is independent and the non-leaves and leaves of type 1 form a vertex cover.*
Parameterized MinLeafOBranching: Results

**Fact**
A normalized out-branching can be found in polynomial time.

**Theorem**
Either $D$ has an OB with at most $n - k$ leaves or $UG(D)$ has a vertex cover of size at most $2k - 3$ (obtained from a normalized OB).

**Corollary**
There is a polytime algorithm that either finds an OB with at most $n - k$ leaves or a tree decomposition of $UG(D)$ of width at most $2k - 3$.

**Theorem**
There is an algorithm of runtime $O(2^{O(k \log k)} + n^2 \log n)$. 