Fast FPT-Approximation of Branchwidth

Fedor V. Fomin and Tuukka Korhonen

UNIVERSITY OF BERGEN
Faculty of Mathematics and Natural Sciences

Results

- Framework for obtaining FPT 2-approximation algorithms for branchwidth of symmetric submodular functions.
- $2^{2^{\mathcal{O}(k)}} \cdot n^2$ time 2-approximation algorithm for rankwidth
  - Improves algorithms parameterized by rankwidth and cliquewidth from $f(k) \cdot n^3$ to $f(k) \cdot n^2$.
- $2^{\mathcal{O}(k)} \cdot n$ time 2-approximation algorithm for graph branchwidth
  - Improves over 3-approximation within the same time.

Algorithm

- Main idea: Iteratively improve branch decomposition by applying **refinement operations**.

Combinatorial Framework:

Let T be a branch decomposition of f and uv an edge of T with $f(uv) = \mathsf{w}(T)$. Now, either

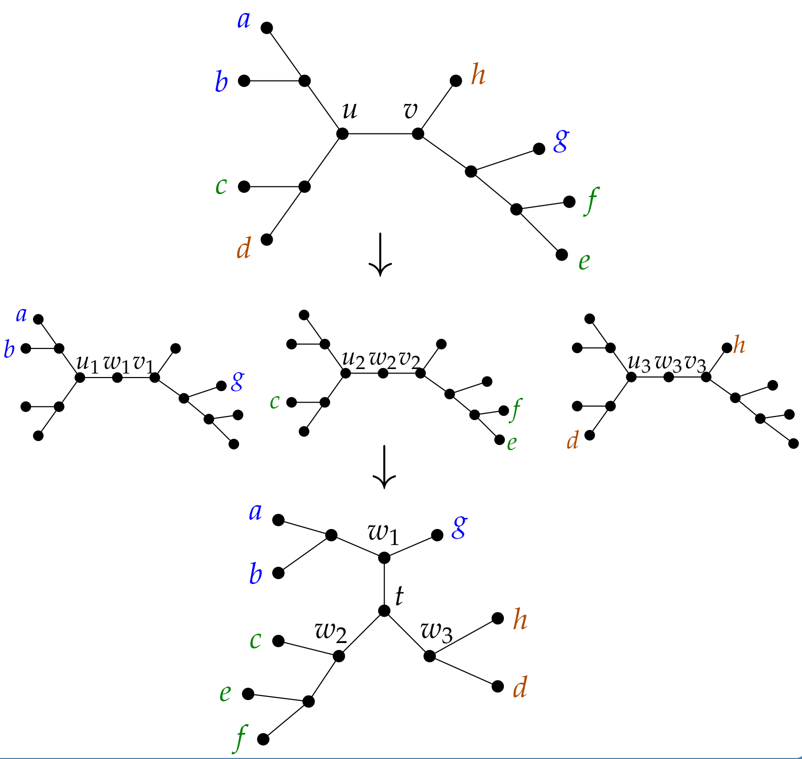
- a refinement operation with (uv, C_1, C_2, C_3), where (C_1, C_2, C_3) is a partition of V, can be applied, either decreasing $\mathsf{w}(T)$ or the number of edges uv with $f(uv) = \mathsf{w}(T)$, or
- $\mathsf{w}(T) \leq 2 \cdot \mathsf{bw}(f)$.

Algorithmic Framework:

If certain dynamic programming operations can be performed in time $t(k)$ per node, then sequence of refinement operations can be done in time $t(k) \cdot 2^{\mathcal{O}(k)} \cdot n$.

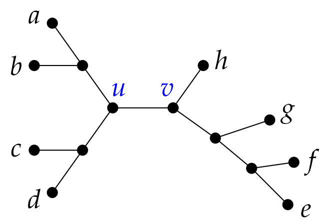
Refinement Operation

Refinement with $(uv, C_1, C_2, C_3) = (uv, \{a, b, g\}, \{c, e, f\}, \{d, h\})$



Definitions

- Let V be a set and $f : 2^V \to \mathbb{Z}_{\geq 0}$ a symmetric submodular function.
- Example of a branch decomposition T of f when $V = \{a, b, c, d, e, f, g, h\}$:



- Denote $f(uv) = f(\{a, b, c, d\}) = f(\{e, f, g, h\})$.
- The width of T is $\mathsf{w}(T) = \max_{uv \in E(T)} f(uv)$.
- The branchwidth $\mathsf{bw}(f)$ of f is the minimum width of a branch decomposition of f.

Rankwidth of a graph G:

$V = V(G)$
$f(U)$ is the GF(2)-rank of $G[U, V(G) \setminus U]$

Branchwidth of a graph G:

$V = E(G)$
$f(U)$ is #vertices incident to both U and $E(G) \setminus U$

Main Ideas

- Refinement that locally improves the decomposition also globally improves the decomposition
- If all leaves of a subtree are inside one part C_i of the refinement partition, then this subtree is not changed
  - number of changed nodes amortizes to $2^{\mathcal{O}(k)} \cdot n$ over the algorithm

Previous Works on Rankwidth

[Oum & Seymour, JCTB'06]: 3-approximation of branchwidth of symmetric submodular functions in $8^k n^{\mathcal{O}(1)}$ time

[Oum, TALG'08]: 3-approximation of rankwidth in $f(k)n^3$ time

[Hliněný & Oum, SICOMP'08]: exact rankwidth in $f(k)n^3$ time