Linear-Time Computation of a Linear Kernel for Dominating Set on Planar Graphs

René van Bevern\(^1\)

joint work with

Sepp Hartung\(^1\), Frank Kammer\(^2\), Rolf Niedermeier\(^1\), and Mathias Weller\(^1\)

\(^1\)Institut für Softwaretechnik und Theoretische Informatik
Technische Universität Berlin, Germany

\(^2\)Institut für Informatik
Universität Augsburg, Germany

Workshop über Algorithmen und Komplexität,
61. Theorietag, Universität Trier
**Dominating Set**

**Input:** A graph $G = (V, E)$ and a natural number $k$.

**Question:** Is there a dominating set $D \subseteq V$ with $|D| \leq k$ for $G$?

Here, $D$ is a dominating set $\iff V \subseteq N[D]$.

Size of a minimum dominating set for $G$ is denoted by $\gamma(G)$.

**Dominating Set** is NP-hard even when restricted to planar graphs.
Example for Dominating Set

The set \( \{v, w\} \) dominates the shown graph.
Problem Kernels

We show an $O(\gamma(G))$-vertex problem kernel for DOMINATING SET on planar graphs that is computable in linear time.

\[ \gamma(G) = \gamma(G') \]

$G$ \quad \text{linear time} \quad G'$

$O(\gamma(G))$ vertices
Why Problem Kernels Matter

In general, problem kernels yield equivalent smaller instances in polynomial time. Hence, used as a preprocessing step, they can

- speed up algorithms that exactly solve NP-hard problems,
- speed up slow but effective approximation algorithms,
- speed up slow but effective kernelization algorithms,
- speed up heuristics often used in practice.

Combining fast kernelization algorithms with effective ones yields small kernels fast.

In contrast, it is unclear how a fast approximation algorithms can be combined with an effective one to get an approximation algorithm that is both fast and effective.
**The Kernel Size Race**

Problem kernels are currently the hottest topic in parameterized algorithmics, resulting in a kernel size race.

**An Example:**

**Feedback Vertex Set:** delete at most $k$ vertices to transform a graph into a forest (make it cycle-free).

Burrage, Estivill-Castro, Fellows, and Langston, IWPEC 2006: $O(k^{11})$-vertex kernel

Bodlaender and van Dijk, TCS 2010: $O(k^3)$-vertex kernel

Thomassé, ACM Trans. Algorithms 2010: $O(k^2)$-vertex kernel
The Kernel Size Race

Problem kernels are currently the hottest topic in parameterized algorithmics, resulting in a kernel size race.

Another Example:

Cluster Editing: delete or add at most $k$ edges to transform a graph into a disjoint union of cliques.

Gramm, Guo, Hüffner, and Niedermeier, TCS 2005: $O(k^2)$-vertex kernel

Fellows, Langston, Rosamond, and Shaw, FCT 2007: $O(k)$-vertex kernel

Guo, TCS 2009: $4k$-vertex kernel

Chen and Meng, COCOON 2010: $2k$-vertex kernel
Problem Kernels for Dominating Set

Focusing on Kernel Size:

Alber, Fellows, and Niedermeier, J. ACM, 2004:
335\(\gamma\)-vertex kernel in \(O(n^3)\) time on planar graphs.

Chen, Fernau, Kanj, and Xia, SIAM J. Comput., 2007:
67\(\gamma\)-vertex kernel in \(O(n^3)\) time on planar graphs.

Focusing on Larger Graph Classes:

Fomin and Thilikos, ICALP 2004:
\(O(\gamma + g)\)-vertex kernel in \(O(gn^3)\) time on graphs of genus \(g\).

Philip, Raman, and Sikdar, ESA 2009:
\(O(\gamma^{2(d+1)^2})\)-vertex kernel in \(O(2^d dn^2)\) time on \(d\)-degenerate graphs.

Fomin, Lokshtanov, Saurabh, and Thilikos, SODA 2010:
\(O(\gamma)\)-vertex kernel in polynomial time on apex-minor free graphs.
Focusing on Running Time:

van Bevern, Hartung, Kammer, Niedermeier, Weller, Manuscript 2011 (submitted):
$O(\gamma)$-vertex kernel in $O(n)$ time.

Small Kernel Fast:

$67\gamma$-vertex kernel in $O(\gamma^3 + n)$ time using the above kernelization as preprocessing step for the following result:

Chen, Fernau, Kanj, Xia, SIAM J. Comput., 2007:
$67\gamma$-vertex kernel in $O(n^3)$ time on planar graphs.
Tools for Obtaining Dominating Set Problem Kernels

Many results exploit a result by Alber, Fellows, and Niedermeier, J. ACM 2004: a graph is decomposable into $O(\gamma)$ regions:

A region $R(v, w)$ between two vertices $v$ and $w$ is a closed bounded subset of the plane such that:

1. the boundary of $R(v, w)$ is formed by two simple paths between $v$ and $w$, each of which has length at most three and
2. all vertices inside the region $R(v, w)$ are from $N[v] \cup N[w]$. 
Problem kernel analysis of Alber et al., J. ACM, 2004

A graph is decomposable into $O(\gamma)$ regions:

Problem kernel is obtained as follows:

1. shrink each region to $O(1)$ vertices and
2. reduce number of vertices outside of regions to $O(\gamma(G))$. 
Our Problem Kernel

Re-use the region framework and data reduction rules by Alber et al. with slight modifications, but find and update the structures to be reduced in linear time.

Example: Finding and Shrinking Regions

Problem: graph is decomposable into $O(\gamma)$ regions, but we do not actually have them. $\Rightarrow$ shrink everything that could be a region.

Alber et al. delete vertices from $N[v] \cup N[w]$ for all vertices $v, w$. $\Rightarrow \Omega(n^2)$ time. Rules applied $O(n)$ times $\Rightarrow O(n^3)$ time.

Assuming that we know how to shrink regions: how to find them?

- Largely exploit the restricted structure of regions and
- know when to stop: do not apply data reduction rules more often than necessary to obtain the problem kernel.
Consider this region $R(v, w)$. 
Assume that $u_1 \in R(v, w)$. Then, $u_1 \in N[v, w]$. 

Assuming $u_1 \in R(v, w)$ means that $u_1$ is in the region $R(v, w)$, and the statement $u_1 \in N[v, w]$ means that $u_1$ is in the neighborhood $N[v, w]$ of $v$ and $w$. The diagram illustrates the relationship between these concepts.
W. l. o. g., assume that $u_1 \in N(w)$. 
Shrinking Regions in Linear Time

If $\deg(u_1) \leq 1$: delete $u_1$ and remember $w$ to be dominating.
Hence, $u_1$ has a neighbor, which, due to planarity, is in $R(v, w)$. 
If $N[N[u_1]] \subseteq N[w]$, delete $u_1$ (remember $w$ to be dominating).
Shrinking Regions in Linear Time

Assume that $u_1$’s neighbor is nonadjacent to $w$. 
Then, $u_1$’s neighbor is adjacent to $v$. 

Shrinking Regions in Linear Time
Observation: inner vertices of $R(v, w)$ lie on short $v$-$w$-paths.
Shrinking Regions in Linear Time

Can quickly list such paths if vertices of $R(v, w)$ have small degree.
If $u_1$ has large degree, then $N(u_1) \cap N(v)$ or $N(u_1) \cap N(w)$ is large.

Shrinking Regions in Linear Time
Assume that $N(u_1) \cap N(v)$ is large $\rightsquigarrow$ red sub-region $R(v, u_1)$. 
Shrinking Regions in Linear Time

Can shrink $R(v, u_1)$ if it only contains vertices with small degree.
Assume that $u_2 \in R(v, u_1)$. By planarity, $|N(u_2) \cap N(1)| \in O(1)$. 

![Diagram of shrinking regions in linear time](image)
Hence, if $u_2$ has large degree, then $N(u_2) \cap N(v)$ (green) is large.
Can shrink $R(v, u_2)$ if it only contains vertices with small degree.
This is the case since vertices in blue regions are safely deletable.
Shrink blue regions \( \leadsto \) vertices in \( R(v, u_2) \) have low degree.
Shrink $R(v, u_2)$ $\leadsto$ vertices in $R(v, u_1)$ have low degree.
Shrinking Regions in Linear Time

Shrink $R(v, u_1) \rightsquigarrow$ vertices in $R(v, w)$ have low degree.
We can now find and shrink $R(v, w)$. 
Three times, basically apply the following algorithm:

1. Preprocess (e.g. degree-one deletion rule)
2. Find deletion candidates regions:
   2.1 For each \( v \in V \), in \( O(\deg(v)) \) time, list all short paths starting at \( v \), having length at most four, and only crossing vertices with constant degree.
   2.2 For each such path \( p \), let \( v_p, w_p \) be its endpoints.
   2.3 If \( p \) contains only vertices of \( N[v_p] \cup N[w_p] \), then \( p \) is a deletion candidate (could be part of a region to shrink).
3. Shrink regions, which contain \( O(n) \) deletion candidates in total, since only \( \sum_{v \in V} \deg(v) = O(n) \) paths are generated.
Conclusion

Not only the size of problem kernels, but also the running time of kernelization algorithms has to be optimized.

Optimize one of size and speed; get the other for free:

- can combine speed-optimized and size-optimized kernelization algorithms to obtain small kernels fast

\[ 67\gamma\text{-vertex kernel in } O(\gamma^3 + n) \text{ time for DOMINATING SET in planar graphs.} \]

\[ \Rightarrow \] faster algorithms for solving hard problems.

\(^1\)Executing the algorithm by Chen et al. (SIAM J. Comput., 2007) after ours