# DAGmaps and Dominance Relationships

Vassilis Tsiaras and Ioannis G. Tollis

Institute of Computer Science, Foundation for Research and Technology-Hellas
{tsiaras,tollis}@ics.forth.gr

## 1 Introduction

In [2] we use the term *DAGmap* to describe space filling visualizations of DAGs according to constraints that generalize treemaps and we show that deciding whether or not a DAG admits a DAGmap drawing is NP-complete. Let $G = (V, E)$ be a DAG with a single source $s$. A component st-graph $G_{u,v}$ of $G$ is a subgraph of $G$ with a single source $u$ and a single sink $v$ that contains at least two edges and that is connected with the rest of $G$ through vertex $u$ and/or vertex $v$. A vertex $w$ *dominates* a vertex $v$ if every path from $s$ to $v$ passes through $w$. The dominance relation in $G$ can be represented in compact form as a tree $T$, called the dominator tree of $G$, in which the dominators of a vertex $v$ are its ancestors. Vertex $w$ is the *immediate dominator* of $v$ if $w$ is the parent of $v$ in $T$. A simple and fast algorithm to compute $T$ has been proposed by Cooper et al. [1]. The *post-dominators* of $G$ are defined as the dominators in the graph obtained from $G$ by reversing all directed edges and assuming that all vertices are reachable from a (possibly artificial) vertex $t$. Using the definition of DAGmaps, it is easy to prove that in a DAGmap of $G$ the rectangle of a vertex $u$ includes the rectangles of all vertices that are dominated (resp. post-dominated) by $u$. Therefore when vertex $u$ dominates vertex $v$ and vertex $v$ post-dominates vertex $u$ then the rectangles $R_u$ and $R_v$ of $u$ and $v$ coincide. Based on this observation, we propose a heuristic algorithm that transforms a DAG $G$ into a DAG $G'$ that admits a DAGmap. When $G$ contains component st-graphs then our algorithm performs significantly fewer duplications than the transformation of $G$ into a tree.

## 2 Transforming a DAG So That It Admits a DAGmap

A component st-graph $G_{u,v}$ of $G$ behaves under vertex duplications as if it is encapsulated in a cluster vertex (c-vertex). When vertex $u$ is duplicated then $G_{u,v}$ is duplicated as a whole. Additionally duplications that start at a vertex $w$ of $G_{u,v}$, where $w \neq u, v$, stop at vertex $v$ (i.e., they do not propagate to the rest of $G$ since $R_v = R_u$). To identify the component st-graphs of $G$ first we compute the dominator and the post-dominator trees of $G$. Then for each non-leaf node $v$ of the post-dominator tree that has more than one incoming edges we find its immediate dominator $u$. The st-graph $G_{u,v}$ that is induced by all paths from $u$ to $v$ in $G$ is a component st-graph of $G$. If the immediate post-dominator of $u$ is $v$

D. Eppstein and E.R. Gansner (Eds.): GD 2009, LNCS 5849, pp. 424–425, 2010.

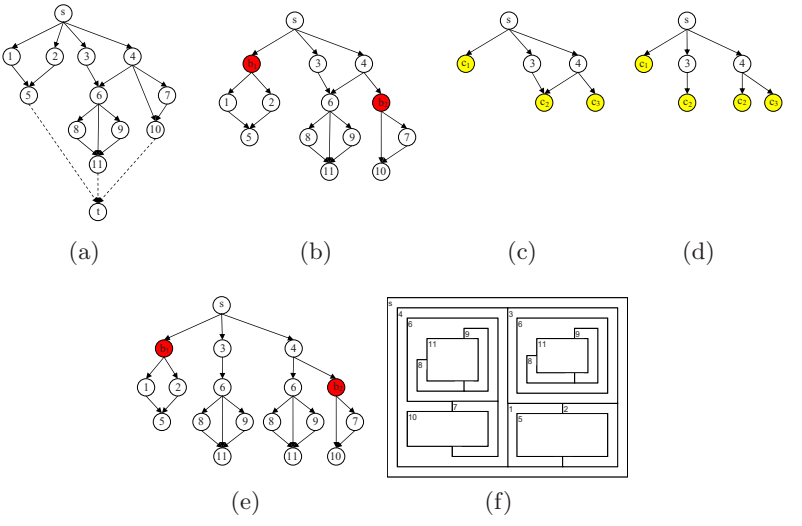


**Fig. 1.** Transforming DAG $G$ into $G'$ that admits a DAGmap. a) A DAG with an artificial sink vertex $t$. b) Introduction of artificial b-vertices. c) Folding of component st-graphs into c-vertices. d) Vertex duplications. e) Unfolding of c-vertices; subgraph $G_{6,11}$ appears twice in DAG $G'$. f) DAGmap drawing of $G'$.

then $G_{u,v}$ is clearly discernible, else it is recommended to introduce an artificial vertex $b$ that is adjacent to $u$ and collects all outgoing edges of $u$ that belong to paths that lead to $v$. It is easy to show that a DAG $G$ is converted into a DAG $G'$ that admits a DAGmap if we first recursively fold the component st-graphs of $G$ and then we repeat until termination the following: unfold a c-vertex and in case that the unfolded subgraph does not admit a DAGmap then perform duplications of its vertices that have more than one incoming edge. Note that if we perform duplications even when the unfolded st-graph admits a DAGmap then an initial st-graph $G$ is converted into a TTSP digraph [2]. In Fig. 1 vertex $s$ dominates vertex 5 but vertex 5 does not post-dominate vertex $s$. Therefore an artificial vertex $b_1$ is introduced. On the other hand the subgraph induced by vertices $s, 3, 4$ and 6 cannot be separated since vertex 4 has an outgoing edge to a vertex that does not belong to the subgraph. Finally c-vertex $c_2$ is duplicated since it has two incoming edges and this leads to duplication of the whole st-graph $G_{6,11}$.

## References

1. Cooper, K.D., Harvey, T.J., Kennedy, K.: A simple, fast dominance algorithm (2001), `http://www.cs.rice.edu/~keith/EMBED/dom.pdf`
2. Tsiaras, V., Triantafilou, S., Tollis, I.G.: DAGmaps: Space Filling Visualization of Directed Acyclic Graphs. JGAA 13(3), 319–347 (2009)