

Minimum Spanning Trees in Weakly Dynamic Graphs - Arbres couvrants minimaux dans les graphes faiblement dynamiques¹

Jean-Yves Colin Moustafa Nakechbandi Hervé Mathieu

Université du Havre, Le Havre, France

June 6, 2014

¹partially supported by GRR Transport, Logistique, Technologies de l'Information/projet APLog, France

The Logistic Problem

- Initially started as a point-to-point Truck Delivery Problem (either as real truck on a road network, or an automated container carrier in a port, or...)
- in a mostly stable environment, except on a few known points (bascule or lift bridge, elevator, ...).

It is now a Minimum Spanning Tree (MST) problem on a dynamic graph.

Models of Dynamic Graphs

Many different models of Dynamic Graphs (with discrete or continuous time)

- Probabilistic models
 - the structure of the problem is fixed, but some parameters are subject to probabilities (the weight of an edge...),
 - or the structure itself (the presence of a node...) is subject to probabilities.

- Time-Varying Graphs

- the structure of the problem is fixed, but some parameters (the weight of an edge...) have several values, each known to be valid at some particular date and not at other moments,
- or the structure itself (the presence of a node...) has several known configurations, each known to be valid at some particular date and not at other moments.

Models of Dynamic Graphs

- Fully Dynamic Graphs
 - the structure of the problem is fixed, but some parameters (the weight of an edge...) may change at any time,
 - or the structure itself (the presence of a node...) may change at any time.
 - the possible operations are usually

- Fully Dynamic Graphs

- the structure of the problem is fixed, but some parameters (the weight of an edge...) may change at any time,
- or the structure itself (the presence of a node...) may change at any time.
- the possible operations are usually
 - remove : remove an edge or a node from the graph,
 - insert : add an edge or a node to the graph,
 - update : modify the parameter of an edge or node.

Minimum Spanning Trees on Dynamic Graphs

Lot of recent studies, on properties of (Fully) Dynamic Graphs

- Shortest paths (Nannicini et Liberti, 2008)
- Connectivity (Holm et al., 2001)

Minimum Spanning Trees on Dynamic Graphs

Lot of recent studies, on properties of (Fully) Dynamic Graphs

- Shortest paths (Nannicini et Liberti, 2008)
- Connectivity (Holm et al., 2001)

More specifically, on the computation of MST on Dynamic Graphs, several solutions are proposed

- using partition and topology trees (Frederickson, 1985, and Frederickson, 1997),
- using sparsification (Eppstein et al, 1996),
- using randomized algorithms (Herzinger et King, 1999),
- using logarithmic decomposition (Holm et al., 2001).

Minimum Spanning Trees on Dynamic Graphs

Almost all proposed solutions follow the same general idea

Minimum Spanning Trees on Dynamic Graphs

Almost all proposed solutions follow the same general idea

- an initial MST is computed for the initial state of the graph,

Minimum Spanning Trees on Dynamic Graphs

Almost all proposed solutions follow the same general idea

- an initial MST is computed for the initial state of the graph,
- an additional structure is added,

Minimum Spanning Trees on Dynamic Graphs

Almost all proposed solutions follow the same general idea

- an initial MST is computed for the initial state of the graph,
- an additional structure is added,
- this additional structure is used to recompute as efficiently as possible the new MST as soon as a change is detected in the dynamic graph,

Minimum Spanning Trees on Dynamic Graphs

Almost all proposed solutions follow the same general idea

- an initial MST is computed for the initial state of the graph,
- an additional structure is added,
- this additional structure is used to recompute as efficiently as possible the new MST as soon as a change is detected in the dynamic graph,
- this structure is usually itself updated too.

These solutions are developed for the most general case of fully dynamic graphs : anything can change, at any time.

Weakly Dynamic Graphs

In our case, only one or a few known locations in the graph may change.
All the others data are stable and will never change.

Weakly Dynamic Graphs

In our case, only one or a few known locations in the graph may change.
All the others data are stable and will never change.

Definition

Weakly Dynamic Graph (Colin et al. 2012).: A Weakly Dynamic Graph is a graph in which there is an one unstable valuated edge (in an undirected graph) or valuated arc (in a directed graph) between two known nodes x_1 and x_2 of the graph. That edge or arc has an unknown a positive value x that may change at any time.

All other edges or arcs are stable and their values never change.

Weakly Dynamic Graphs

In our case, only one or a few known locations in the graph may change. All the others data are stable and will never change.

Definition

Weakly Dynamic Graph (Colin et al. 2012).: A Weakly Dynamic Graph is a graph in which there is an one unstable valuated edge (in an undirected graph) or valuated arc (in a directed graph) between two known nodes x_1 and x_2 of the graph. That edge or arc has an unknown a positive value x that may change at any time.

All other edges or arcs are stable and their values never change.

For the Single Source Shortest Path Problem, see (Ahuja et al., 2008) and (Colin et al. 2012).

Weakly Dynamic Graphs

In our case, only one or a few known locations in the graph may change. All the others data are stable and will never change.

Definition

Weakly Dynamic Graph (Colin et al. 2012).: A Weakly Dynamic Graph is a graph in which there is an one unstable valuated edge (in an undirected graph) or valuated arc (in a directed graph) between two known nodes x_1 and x_2 of the graph. That edge or arc has an unknown a positive value x that may change at any time.

All other edges or arcs are stable and their values never change.

For the Single Source Shortest Path Problem, see (Ahuja et al., 2008) and (Colin et al. 2012).

And about Minimum Spanning Trees?

The general MST problem

Start with a more general problem. Let $G = (V, E)$ be a simple undirected graph, with V being the set of vertices, and E being the set of edges of G .

The general MST problem

Start with a more general problem. Let $G = (V, E)$ be a simple undirected graph, with V being the set of vertices, and E being the set of edges of G .

Definition

An *Edge-Constrained Spanning Tree* $\text{ECST}(V, E, E^+, E^-)$ of an undirected graph $G = (V, E)$ is a spanning tree of G with E^+ and E^- being two disjoint subsets of E , and such that all edges in E^+ are in this spanning tree and no edge of E^- is in this spanning tree.

The general MST problem

In the following, we will call E^+ the set of *mandatory* edges, and E^- the set of *forbiddden* edges.

Some remarks:

- this definition of mandatory is different from the definition of *mandatory* for an edge in MST in other papers, where an edge is said to mandatory if all spanning trees must include it, else this ST will not be minimal,
- the term *constrained* applied to a spanning tree has also the different meaning in other papers that some global criterion must be satisfied by the tree, such as a maximal weight, diameter or degree.

The general MST problem

- Of course, an edge-constrained spanning tree as we defined it may not always exist. For example, the subset E^+ of *mandatory* edges may already include a cycle, or the subset E^- of *forbidden* edges may be such that the graph without its edges is not a connected graph anymore.

Without loss of generality, we will suppose in the rest of this paper that one edge-constrained spanning tree actually exists.

The general MST problem

An *edge-weighted edge-constrained minimal spanning tree problem*

$P = (V, E, w, E^+, E^-)$, is a problem in which

V is a set of vertices,

E is a set of edges $(i, j) \mid i \in V, j \in V$,

w is a weight function with $w : E \rightarrow \mathbb{R}$,

E^+ is a subset of E of *mandatory* edges that must belong to the tree,

E^- is a subset of E of *forbidden* edges that are not allowed to belong to the tree.

The general MST problem

Definition

A spanning tree T in an edge weighted edge-constrained minimum (resp. maximum) spanning tree problem $P = (V, E, w, E^+, E^-)$ is an *edge-constrained spanning tree* of $G = V, E$ that verifies $\text{ECST}(V, E, E^+, E^-)$ and such that no other spanning tree that verifies $\text{ECST}(V, E, E^+, E^-)$ has a lower (resp. higher) weight.

The general MST problem

- A solution of the classical minimum (resp. maximum) spanning tree problem is then a solution of the edge weighted edge-constrained minimum (resp. maximum) spanning tree problem $P = (V, E, w, \emptyset, \emptyset)$.

The general MST problem

- A solution of the classical minimum (resp. maximum) spanning tree problem is then a solution of the edge weighted edge-constrained minimum (resp. maximum) spanning tree problem $P = (V, E, w, \emptyset, \emptyset)$.
- Computing a minimum or maximum edge-weighted edge-constrained spanning tree without the forbidden edges of E^- is trivial. Just build a subgraph of $G = (V, E)$ without these edges and apply an algorithm such a Prim algorithm, or Kruskal algorithm (or any other).

The general MST problem

- A solution of the classical minimum (resp. maximum) spanning tree problem is then a solution of the edge weighted edge-constrained minimum (resp. maximum) spanning tree problem $P = (V, E, w, \emptyset, \emptyset)$.
- Computing a minimum or maximum edge-weighted edge-constrained spanning tree without the forbidden edges of E^- is trivial. Just build a subgraph of $G = (V, E)$ without these edges and apply an algorithm such a Prim algorithm, or Kruskal algorithm (or any other).
- Computing a minimum or maximum edge-weighted edge-constrained spanning tree in a problem P with mandatory edges is more difficult.

The general MST problem

First suppose that E^+ has only one edge (i, j) . A modified Prim algorithm may then be used. In this algorithm, instead of starting from a random vertex, and without any initial edge in the tree, we may start with the initial set of vertices $\{i, j\}$ and with the initial edge (i, j) in the tree and apply Prim algorithm from there.

The general MST problem

First suppose that E^+ has only one edge (i, j) . A modified Prim algorithm may then be used. In this algorithm, instead of starting from a random vertex, and without any initial edge in the tree, we may start with the initial set of vertices $\{i, j\}$ and with the initial edge (i, j) in the tree and apply Prim algorithm from there.

Theorem

In an edge-weighted edge-constrained spanning tree problem $P = (V, E, w, E^+, E^-)$, if E^+ has only one edge, then the modified algorithm of Prim that starts from edge (i, j) of E^+ computes a minimum edge-constrained spanning tree.

The general MST problem

Now suppose that E^+ has two edges or more. The modified Prim algorithm cannot be used.

We propose the following modified Kruskal algorithm. Instead of starting from an empty subset of G that will slowly be grown into a spanning tree, we may start with the initial subset E^+ and apply Kruskal algorithm from there.

The general MST problem

Now suppose that E^+ has two edges or more. The modified Prim algorithm cannot be used.

We propose the following modified Kruskal algorithm. Instead of starting from an empty subset of G that will slowly be grown into a spanning tree, we may start with with the initial subset E^+ and apply Kruskal algorithm from there.

Theorem

In an edge-weighted edge-constrained spanning tree problem $P = (V, E, w, E^+, E^-)$, if E^+ has one edge or more, then the modified Kruskal algorithm that starts with all edges (i, j) of E^+ computes a minimum edge-constrained spanning tree.

MST and Weakly Dynamic Graphs

We can now use the above result on a Weakly Dynamic Graph with one non stable edge.

- first solve the minimum edge-constrained spanning tree problem $P = (V, E, w, \emptyset, \{(i, j)\})$, i.e. the problem without the non stable edge. Its value is the sum of its edges d_s and is a constant.
- next solve the minimum edge-constrained spanning tree problem $P = (V, E, w, \{(i, j)\}, \emptyset)$, i.e. the problem with the mandatory non stable edge. Its value is the sum of its edges d_v which is the sum of its stable edges, plus the non stable value x , that may change at any time.

MST and Weakly Dynamic Graphs

By comparing the values of d_s and d_v , we can deduce the *critical value* $cv(x)$ of x .

Definition

The *critical value* $cv(x)$ of x is the value of x such that the minimum spanning tree will be either the first tree built (if $x > cv(x)$) or the second one (if $x < cv(x)$).

Some Comments

- Because everything is precomputed, the two trees built, and the critical value $cv(x)$ may be stored somewhere and no recomputation is needed each time the non stable changes
 - either the new value of x is on the same side of the critical value $cv(x)$ than the old value of x , and no change is needed,
 - or the new value of x is on the other side of the critical value $cv(x)$ than the old value of x , and the precomputed other tree will be used instead of the current one.

The response time is the best possible.

Some Comments

- These result may be extended to a small number of non stable edges, but the combinatorial nature of this problem makes it impractical for even a medium number of non stable edges,

Some Comments

- These result may be extended to a small number of non stable edges, but the combinatorial nature of this problem makes it impractical for even a medium number of non stable edges,
- however, if we suppose that only one change is possible at any time on one of several non stable edges, and that the delay between two changes is *long enough*, then after each change we can solve m (m being the number of non stable edges), separate problems (with the corresponding trees and critical values for each non stable edge) and have the best possible response time when the next change occurs.

Some Comments

- These result may be extended to a small number of non stable edges, but the combinatorial nature of this problem makes it impractical for even a medium number of non stable edges,
- however, if we suppose that only one change is possible at any time on one of several non stable edges, and that the delay between two changes is *long enough*, then after each change we can solve m (m being the number of non stable edges), separate problems (with the corresponding trees and critical values for each non stable edge) and have the best possible response time when the next change occurs.
- We are currently trying to improve these precomputations.

Thank You