Heuristic Algorithms using Tree Decompositions

Report 6 17/11/2021

De afgelopen weken was ik vooral bezig met implementatie. Eerst heb ik de structuur van een tree decomposition wat verder uitgewerkt door aparte klasses aan te maken voor elk type bag. Ook de solver structuur heb ik wat aangepast/uitgebreid.
Vervolgens ben heb ik (een basic versie) van het algoritme geïmplementeerd. Zoals vorige keer besproken zal ik verschillende kleuringen tegelijkertijd bijhouden ipv slechts één. Stel dat dit aantal *n* is, dan is het niet zo dat er exact *n* kleuringen zijn, maar ten hoogste *n* kleuringen. Momenteel doet het algoritme volgende dingen bij elk type bag:

- **Leaf.** Geef gewoon de initiele kleuring terug.
- **Forget vertex.** Zal de kleuringen terug geven die gevonden werden bij de child node.
- **Introduce vertex v.** Het algoritme bekijkt elke kleuring van de child node en zal vertex v elk mogelijk kleur geven, waarvan enkel de *n* beste kleuringen bijgehouden worden. Hier geef ik v elk mogelijk kleur ipv enkel maar de kleuren van de buren omdat het mogelijk is dat initieel een verkeerde kleur was gegeven aan een bepaalde vertex, waardoor deze fout propageert naar de buur vertices en zo verder.
- **Join.** Elk paar kleuringin waarbij de eerste kleuring van het linker kind komt en de tweede kleuring van het rechter kind wordt bekeken en vervolgens worden de vertices één voor één (volgens id op dit moment) gekleurd als volgt: indien de vertex in geen van beide kleuringen een kleur heeft, dan zal de uiteindelijke vertex geen kleur krijgen, indien de vertex in één van de kleuringen een kleur heeft krijgt deze dat kleur, en indien de vertex in beide kleuringen een kleur heeft, dan zal het beste kleur gekozen worden.

Ik heb al enkele testen gedaan, en het algoritme is nog niet competitief met greedy mhv of growth mhv. Er zijn wel nog verschillende optimalisaties mogelijk. Het zou bijvoorbeeld mogelijk zijn om ook te zorgen dat de berekende kleuringen in een node zo veel mogelijk verschillen. Op die manier zal het algoritme op een groter deel van de search space kunnen focussen. Ook zou het misschien handig zijn om het aantal kleuringen dat bijgehouden worden wat meer flexibel te maken. Als er bijvoorbeeld *n*+1 kleuringen zijn met zelfde aantal happy vertices, dan zal een willekeurige kleuring verwijderd worden. Ook zal ik nog een beter volgorde bepalen voor het kleuren van de vertices bij de join node.