Heuristic Algorithms using Tree Decompositions

Report 5 3/11/2021

Zoals ik zei op onze vorige meeting heb ik eerst gefocust op wat verdere implementatie. Ondertussen heb ik een structuur voor MHV solvers en ook al een implementatie van greedy MHV en growth MHV (growth MHV kan wel nog geoptimaliseerd worden, maar dit is geen prioriteit op dit moment).

Verder heb ik ook nog een geprobeerd om het idee van Jorik verder uit te werken. Hierbij heb ik vooral het probleem dat ik niet weet hoe ik de solutions van child nodes kan gebruiken in de solution van een nieuwe node.

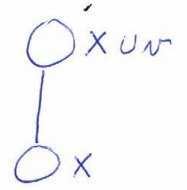
Ik heb ook nog gedacht aan een andere, gelijkaardige strategie die werkt op een nice tree decomposition (dus met introduce vertex, forget vertex en join bags). Het idee is om bij elke bag exact 1 colouring bij te houden voor de vertices in de subtree. Bij elk type zijn dan de volgende operaties nodig.

- **Introduce vertex node.** De nieuwe vertex moet een kleur aangewezen krijgen. Hiervoor kun alle mogelijke kleuren proberen en kijken welke het meeste happy vertices geeft (optimalisatie kan zijn om enkel maar de kleuren van de neighbours te bekijken). Dit geeft O(k) kleuringen die gecheckt moeten worden.
- **Forget vertex node.** Bij deze node moet er niets gebeuren want de ‘vergeten vertex’ zal gewoon het kleur krijgen van de kleuring behorende bij de vorige bag.
- **Join node.** Twee kleuringen van een set vertices moeten gemerged worden. Mijn idee zou dan zijn om de vertices in de bag één voor één te bekijken en dan checken welk kleur voor zorgt het meeste happy vertices. Hierbij is de volgorde uiteraard een belangrijk aspect en kun je volgens mij verschillende strategien gebruiken zoals *largest degree* of *connected to most coloured vertices* of *introduced in bag longest ago*.
-

Volgens mij zou het ook nog goed kunnen zijn om regelmatig de kleuring in een bag te optimaliseren. Dit zou dan gelijkaardig zijn aan de strategie bij de join node, maar je hebt slechts één kleuring en probeert alle andere kleuren. Dit zou goed zijn omdat anders het kleur van een vertex relatief fixed (kan enkel wijzigen bij join nodes en dan enkel maar naar keuze tussen 2 kleuren). Ik weet niet hoeveel effect dit zou hebben maar experimenten zouden dit kunnen tonen. Ook moet ik dan testen hoe vaak een ‘herkleuring’ zou gebeuren (eg na elke bag, om de x bags, misschien enkel bij forget bags).

Hieronder heb ik nog een iets grafischer manier/korte samenvatting geplaats van wat er bij elk type node zou gebeuren tijdens het algoritme. De uiteindelijke kleuring is dan de kleuring verkregen bij de root node.

· introduce vertex

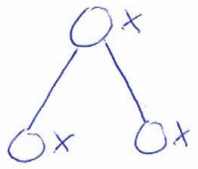


→ colour v in all k colours and check which gives most happy vertices

forget vertex

→ nothing needs to happen

join



colourings: $[c_{1,\ell}, \dots, c_{t,\ell}]$ for left child
$[c_{1,r}, \dots, c_{t,r}]$ for right child

for i in $1..t$
: c = best colour of $c_{i,\ell}$ and $c_{i,r}$
: colour vertex i in parent in colour c

↳ order is important