Heuristic Algorithms using Tree Decompositions

Report 9 22/12/2021

Ik heb eerst en vooral nog wat verder gewerkt om nieuwe handlers te maken voor join nodes en heb er twee toegevoegd:

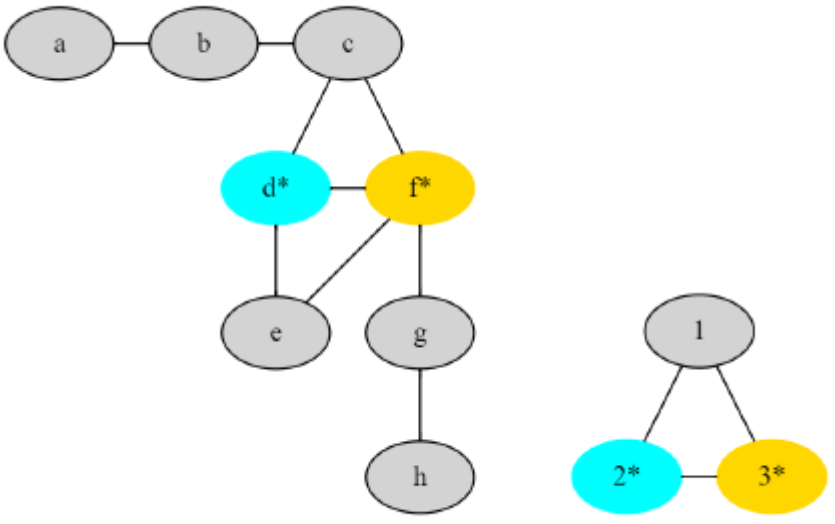
- Voor elk paar kleuringen van het linker en rechter kind, maak twee kleuringen waarin je in de eerste kleuring de vertices in de bag kleurt volgens het linker kind en in de andere volgens het rechter kind.
- Een greedy MHV gebaseerde approach, waarin ik alle vertices in de bag het beste kleur geef.

Ook zoals de laatste handler kan ik één maken die gebaseerd is op growth MHV. Het probleem dat ik hier wel bij heb is dat ik eigenlijk geen rekening houd met de kleuren van de vertices in de bag die reeds berekend werden.

Verder heb ik ook nog een 'nieuwe' manier toegevoegd om een paritele kleuring te evalueren. Voordien als ik naar de potentieel happy vertices keek, dan keek ik enkel naar de gekleurde. Ik kan echter ook kijken naar de vertices die niet gekleurd zijn, wat deze nieuwe methode doet. Dit is echter veel trager in de huidige implementatie.

Ook ben ik bezig met een herstructurering van de code, met als doel backpointers toe te voegen aan het algoritme. Dit is nog niet volledig omdat ik nog enkele details moet uitwerken. Één van de redenen dat het algoritme zo traag was is omdat ik de evaluatie van een partiele oplossing volledig van scratch herbereken. Dit zou in de nieuwe implementatie niet meer nodig zijn.

Ik probeer de structuur zo te doen zodanig dat ik eenvoudig referenties kan toevoegen. Dit idee werd al besproken in een paper van Thiruvady, maar is verder uitgewerkt in de thesis van Florian Peeters. (https://github.com/LouisCarpentier42/HeuristicAlgorithmsUsingTreeDecompositions/blob/main/documents/literature/maximumHappyVertices/OplossingsmethodesVoorHetMaximumHappyVerticesProbleem.pdf, hoofdstuk 4). Het basic idee is dat sommige vertices altijd het zelfde kleur moeten hebben als een andere (eventueel ongekleurde) vertex. Dan kun je een referentie bijhouden naar die vertex en moet je geen tijd besteden aan het kleuren van de vertex en kleur je deze pas achteraf. De figuur hieronder toont dit voorbeeldje, waarin bijvoorbeeld vertex a hetzelfde kleur als vertex b moet hebben, en vertex b hetzelfde kleur als vertex c. Hieruit volgt dat je deze kunt reduceren uit de graaf. Dit zou een algemene optimalisatie geven voor alle algoritmes.



(a) Originele graaf (b) Gereduceerde graaf