RWTH Aachen **WS 2011/12**
Lehrgebiet Theoretische Informatik Tutorial 12
Langer—Reidl—Rossmanith—Sikdar 03.02.2012

Parameterized Algorithms Tutorial

Tutorial Exercise T31

The problem Red-Blue Dominating Set is defined as follows: given a graph G whose vertex set is partitioned into two color sets $R \uplus B = V(G)$ and an integer k, decide whether there exists a set of k red vertices that dominate all blue vertices.
Express the above problem as a $\mathrm{MSO}_1[L]$ formula, where $L = \{\text{red,blue}\}$ is the set of allowed labels. Can this problem be expressed in MSO_2 *without* labels? What does this mean for the logics $\mathrm{MSO}_1[L]$ and MSO_2?

Proposed Solution

we can formalize Red-Blue Dominating Set as follows in $\mathrm{MSO}_1[L]$:

$$\exists D(\forall x(x \in D \to red(x)) \land \forall y(blue(y) \to \exists x(adj(x,y) \land x \in D)))$$

This problem cannot be stated in MSO_2 without labels, which means that $\mathrm{MSO}_1[L]$ is not a strictly weaker logic than MSO_2 (which holds for MSO_1).

Tutorial Exercise T32

Recall the definition of MSO_1 and MSO_2. In the following, let G be a graph and A be some vertex subset of G. Which one of the following properties are expressible in either logic?

1. A forms a cycle
2. A forms an induced cycle
3. G has some hereditary graph property $\mathcal{P}$ characterized by a finite set of forbidden subgraphs
4. G has some hereditary graph property $\mathcal{P}$ characterized by an infinite set of forbidden subgraphs

What consequence do items 1 and 2 have for Hamiltonian Cycle?

Proposed Solution

While an induced cycle can be expressed in both MSO_1 and MSO_2, a cylce is not expressible by MSO_1. The intuition here is that while it can easily be expressed that each vertex in an induced cycle has exactly two neighbours and no more, it is impossible to allow other neighbours on the cycle *and* still validate that the structure indeed is a cycle. This implies that Hamiltonian Cycle can be formalized in MSO_2 but not in MSO_1.
Hereditary graph properties characterized by a finite forbidden set are clearly expressible in MSO_1 as we can easily define a formula which models the existence of a fixed subgraph. For properties characterized by an *infinite* forbidden set this is clearly not possbile for MSO_1, as we can define the forbidden set to be some non-expressible set of graphs (e.g. the set of all cycle that have even length). For MSO_2 this question is open.

Tutorial Exercise T33
Develop dynamic programming algorithms for the following problems on graphs of bounded treewidth:

- 3-Colorability
- Triangle Packing (vertex disjoint)

State the runnning time of each algorithm.

Proposed Solution
The 3-Colorability works similar to the dynamic programming for Vertex Cover, but with three states per vertex.
For Triangle Packing we assign a state to each *edge* of a bag. The states needed are

0 : This edge is not part of any packing triangle

1 : This edge is part of a packing triangle which has not been seen completly

2 : This edge is part of a packing triangle which has been seen completly

If we introduce a vertex v, we assign all cominbations of states to the edges in the current bag that are incident to v, a forget works as always by collapsing equivalent configurations. The following constraints apply:

- A configuration that assigns state 2 to all edges of a triangle in the current bag must assigne state 0 to all further edges incident to that triangle
- A configuration that assigns state 1 to two edges of a triangle and 0 or 2 to the remaning one is invalid
- A configuration that assignes state 1 to an edge that is subsequently forgotten is invalid after that forget step
- If an edge of state 2 is forgotten and the corresponding packing triangle is entirely contained in the bag *before* the forget stept we increase the counter in the subsequent configuration

A join-stept works by taking the maximum of compatible entries from the two tables. An edge with state 0 in one configuration must have state 0 in the other, too; an edge with state 1 is compatible to an edge of state 2 and is mapped to an edge of state 2 in the join-configuration. Two edges of state 2 are *not* compatible.