Theo III - Blatt 4

Benjamin Möller & Nick Daiber

December 8, 2024

3

a

Sei ein Graph $G(V, E)$, $c : E \to \mathbb{R}$, sowie $r : E \to \mathbb{R}$ und s, t gegeben. Um CSP auf das Rucksackproblem zu reduzieren geben wir jedem Knoten $v \in V$ mit kanten (u, v) einen Wert $w(v) = \frac{1}{\min\{u+c(u,v)\}}$. Somit sind nahe knoten mehr wert als weit entfernte. Danach hat jeder Knoten ein gewicht $w(v) = R(v)$. Da das Rucksackproblem NP-Hart ist, ist auch CSP NP-Hart

b

```
func CSP(graph G(V, E), source, destination, R) {
    // this holds the shortest from the source to
    // node v with constraint ≤R with all initially at ∞
    min_distance[v][R] = {{∞, ∞, ...}, ...};

    min_distance[source][0] = 0;

    // iterating over all constraints
    for k = 0 to R {
        for u in V {
            // checking if u is reachable
            if (min_distance[u][k] < ∞) {
                // checking all outgoing verteces
                for each outgoing edge v {
                    weight = w(u, v)
                    // reassign if lower
                    if (k + weight ≤ R) {
                        min_distance[v][k + weight] = min(
                            min_distance[v][k + weight],
                            min_distance[u][k] + weight
                            )
                    }
                }
            }
        }
    }

    // checking shortest path for all constraints
    for k = 0 to R {
        result = min(result, min_distance[target][k])
    }


    if (result == ∞)
        return NO_PATH_FOUND
    else
        return result
}
```