Algorithmen und Komplexität

Greedy-Algorithmen – Lösungen

11. November 2013

1. a) Greedy-Algorithmus:

g_i	in den Rucksack?	Gewicht Rucksack	Wert Rucksack
17	passt rein	17	31
16	passt rein	33	54
14	passt nicht rein	33	54
12	passt nicht rein	33	54
9	passt nicht rein	33	54
3	passt rein	36	59
2	passt rein	38	61

b) Bruchteilrucksack:

i	1	2	3	4	5	6	7
w_i	31	23	17	13	8	5	2
g_i	17	16	14	12	9	3	2
$\frac{w_i}{g_i}$	1,824	1,438	1,214	1,083	0,889	1,667	1,000

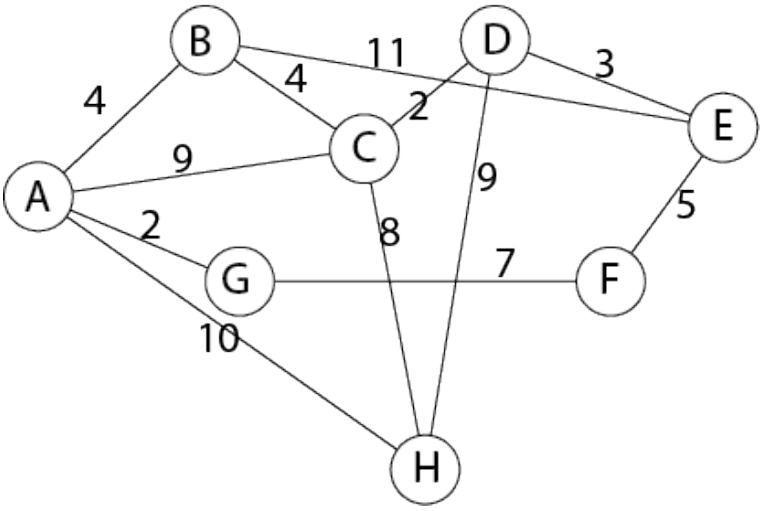
Dadurch ergibt sich eine neue Reihenfolge:

i	1	2	3	4	5	6	7
w_i	31	5	23	17	13	2	8
g_i	17	3	16	14	12	2	9
$\frac{w_i}{g_i}$	1,824	1,667	1,438	1,214	1,083	1,000	0,889

$$K = 17 + 3 + 16 + \frac{4}{14} = 40$$

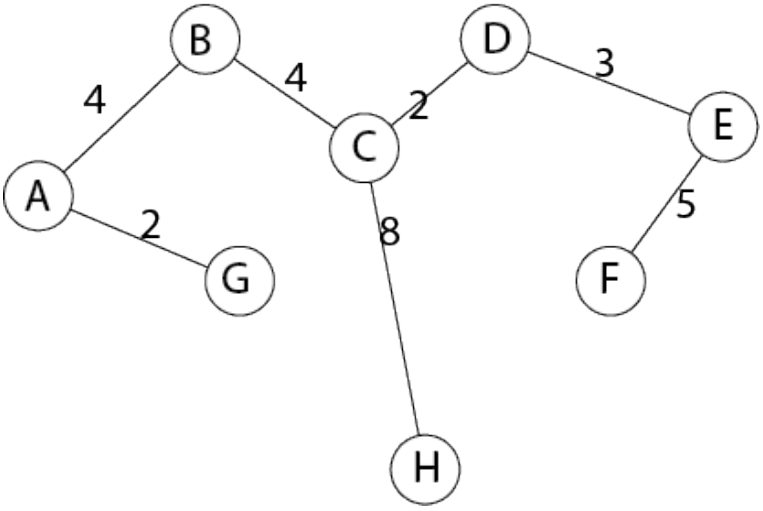
$$W = 31 + 5 + 23 + \frac{4}{14} \cdot 17 = 63,857$$

2. Graph:



Der kürzeste Weg von **A** nach **E** geht über A → B → C → D → E mit einer Gesamtlänge von 13.

3. Minimaler Spannbaum:



4. Umsetzung der Geldwechselaufgabe in PHP

```php
<?php

abstract class GreedyTemplate
{
    protected $kandidatenliste;
    protected $resultatliste;

    abstract protected function loesung();
    abstract protected function okay(Array $neues_resultat);
    abstract protected function naechster();
    abstract protected function modifyPlus($kandidat);
    abstract protected function modifyMinus($kandidat);
    abstract protected function ziel();

    public function helfer()
    {
        if ($this->loesung()) {
            $this->ziel();
            exit();
        } elseif (0 == count($this->kandidatenliste) && 0 == count($this->
            resultatliste)) {
            echo "Es gibt keine Loesung!\n";
            exit();
        } else {
            $neuer_kandidat = $this->naechster();

            $neues_resultat = $this->resultatliste;
            $neues_resultat[] = $neuer_kandidat;

            if ($this->okay($neues_resultat)) {
                $this->modifyPlus($neuer_kandidat);
                $this->resultatliste = $neues_resultat;
                $this->helfer();
            } else {
                $this->modifyMinus($neuer_kandidat);
                $this->helfer();
            }
        }
    }
}

class Geldwechsel extends GreedyTemplate
{
    protected $restbetrag;

    public function __construct()
    {
        $this->resultatliste = [];
    }

    public function setKandidatenliste(Array $kandidaten)
    {
        $this->kandidatenliste = $kandidaten;
    }

    public function setRestbetrag($restbetrag)
    {
        $this->restbetrag = $restbetrag;
    }

```

```
60      protected function loesung()
61      {
62          return ($this->restbetrag == array_sum($this->resultatliste));
63      }
64
65      protected function okay(Array $neues_resultat)
66      {
67          return (array_sum($neues_resultat) <= $this->restbetrag);
68      }
69
70      protected function naechster()
71      {
72          return array_shift($this->kandidatenliste);
73      }
74
75      protected function modifyPlus($kandidat)
76      {
77          array_unshift($this->kandidatenliste, $kandidat);
78      }
79
80      protected function modifyMinus($kandidat)
81      {
82
83      }
84
85      protected function ziel()
86      {
87          echo "(" . implode(" ", $this->resultatliste) . ") " . count($this->
                resultatliste) . "\n";
88      }
89  }
90
91  $geld = new Geldwechsel();
92  $geld->setKandidatenliste([200, 100, 50, 20, 10, 5, 2, 1]);
93  $geld->setRestbetrag(1764);
94  $geld->helfer();
```

Die Funktion *modifyMinus* ist in dieser Lösung ohne konkrete Implementierung, da die verwendete PHP-interne Funktion *array_shift* bereits das erste Element aus dem Array entfernt. Aus diesem Grund wird es bei *modifyPlus* wieder eingefügt.