Wiskunde 13/14

Jan van Hulzen / Liv Harkes

September 23rd 2012 version 1.0

Optimization : Les 4

Organisatie van de cursus

Week	50 min	100 min	100 min	
1	HC	PR	ZS	Opdracht 1
2	HC	PR	ZS	
3	HC	PR	ZS	Opdracht 2
4	HC	PR	ZS	
5		PR	ZS	Opdracht 3
6		PR	ZS	
7		PR	ZS	

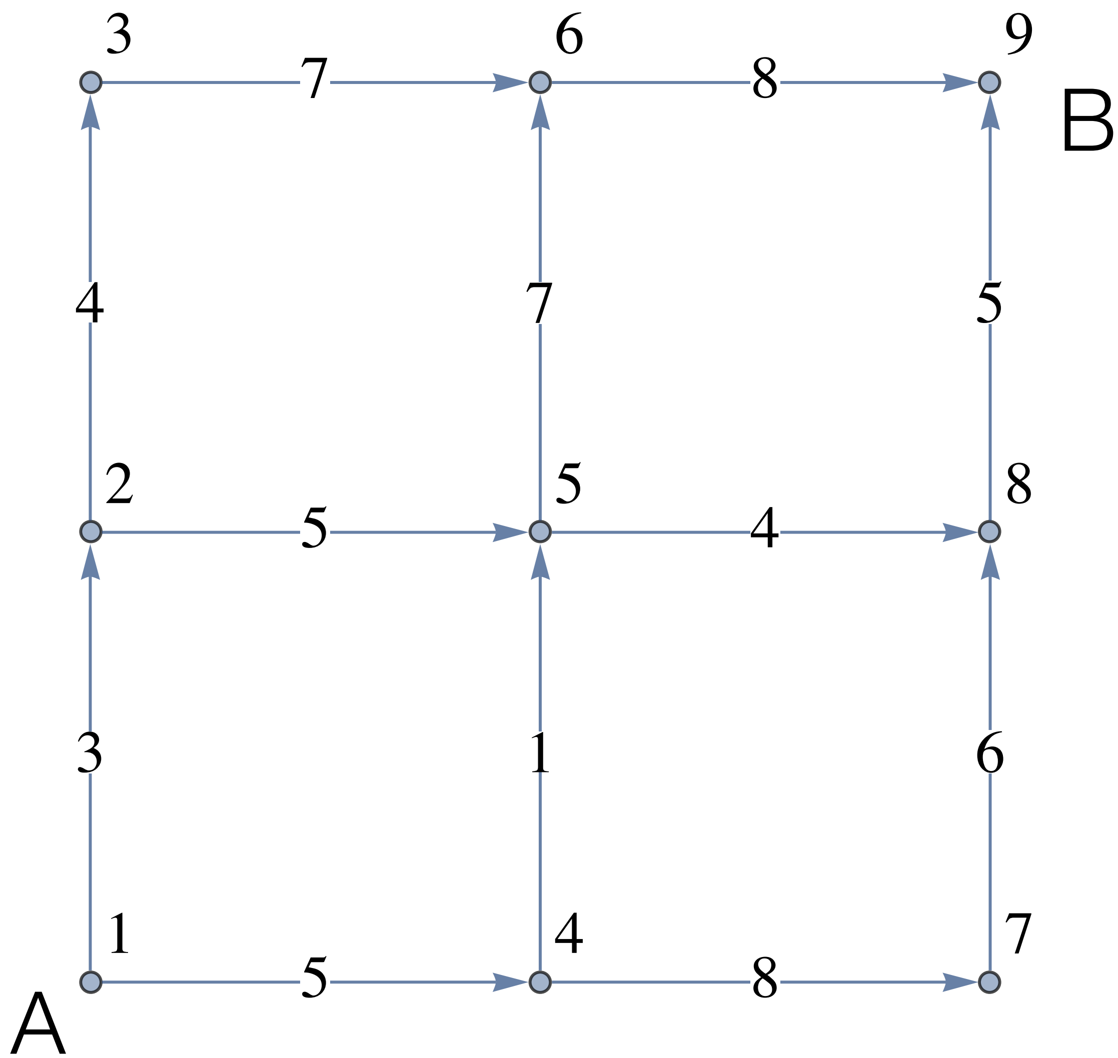
Dynamic Programming

- Dynamic Programming
  - Voorbeeld #1, Korste pad probleem met Manhattan netwerk
  - Voorbeeld #2, Korste pad probleem
  - Voorbeeld #3, Voorraad beheer plobleem
  - Voorbeeld #4, Traveling salesman

Voorbeeld #1 : Kortste pad

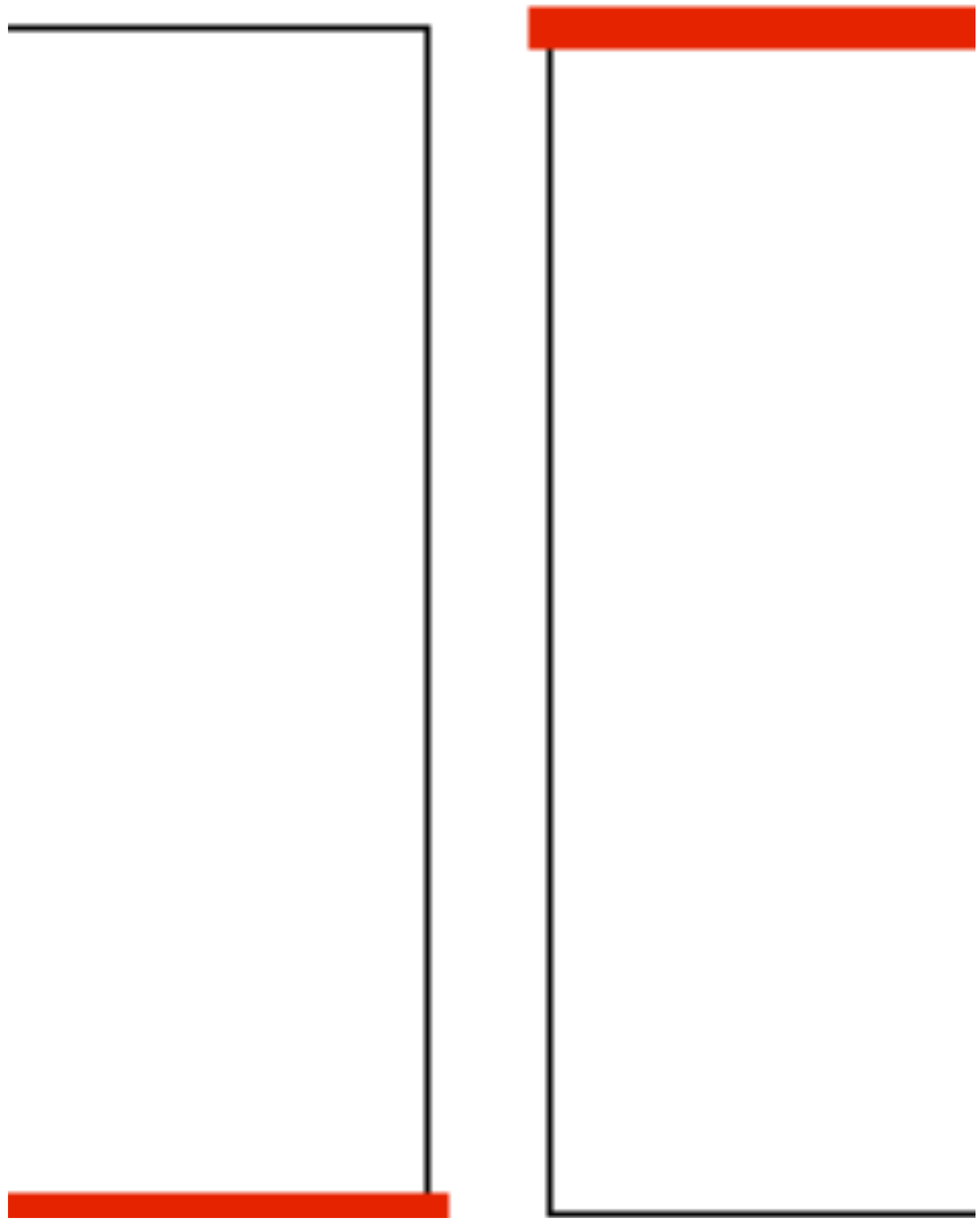
Dynamic Programming

- Korste pad van A naar B



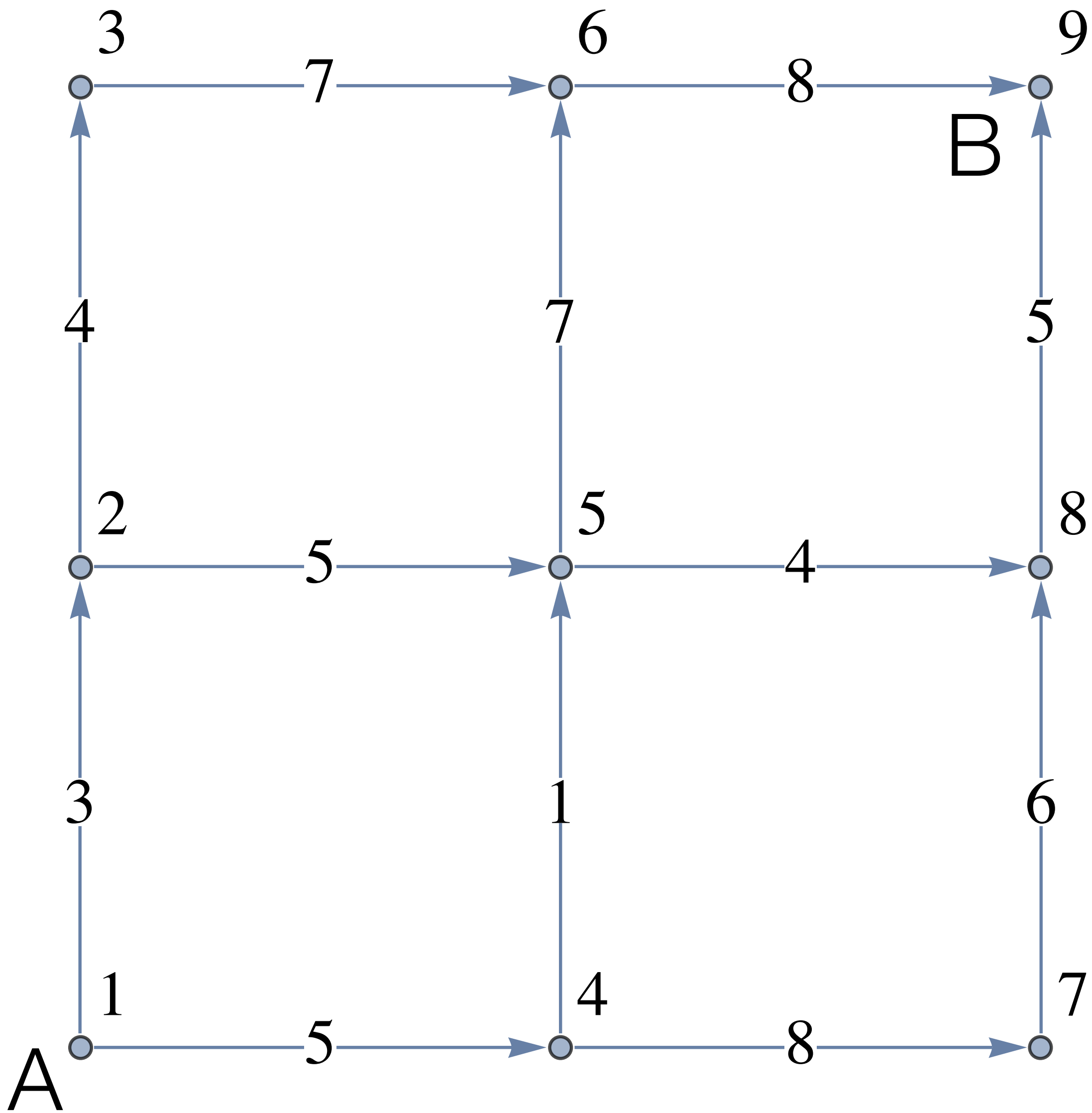
Dynamic Programming : Brute force

- 1x1 grid, 2 paden
- 2x2 grid, 6 paden
- 3x3 grid, 20 paden
- 4x4 grid, 70 paden
- 5x5 grid 252 paden
- nxn grid $\frac{2n!}{n!n!}$ paden
- Benadering $\sqrt{n/\pi}\,2^{2n+1}$
    - bij 10^8 berekeningen per seconde
    - n=25, 2 jaar rekenen
    - n=30, 2000 jaar rekenen



Dynamic programming

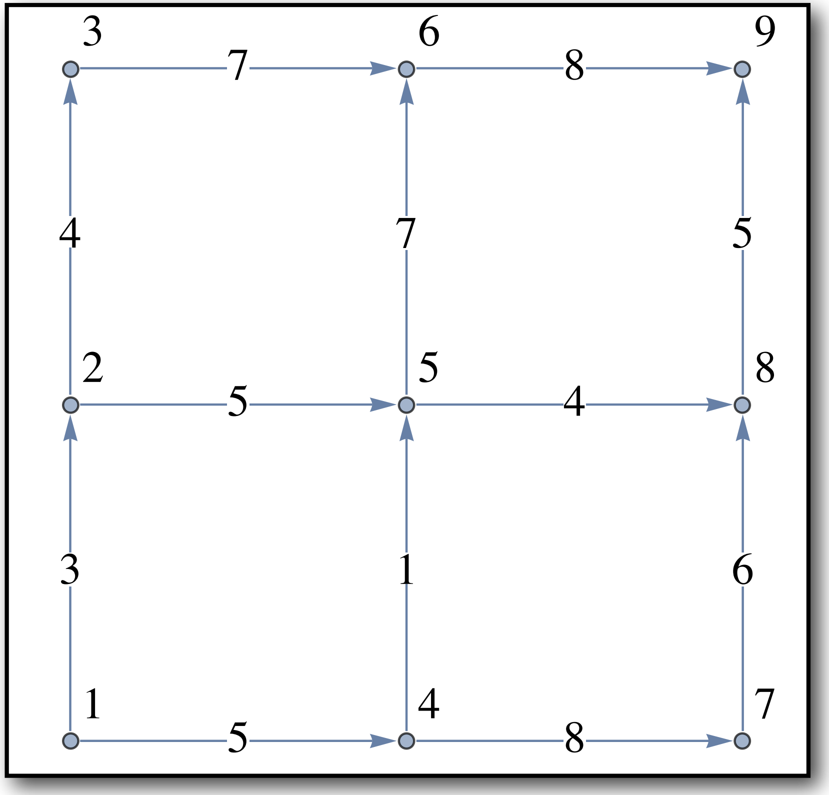
- Kortste pad van A naar B
  - Adjacency matrix
  - Recursief algoritme



Dynamic Programming

- Adjacency matrix

$$
c = \begin{pmatrix}
\infty & 3 & \infty & 5 & \infty & \infty & \infty & \infty & \infty \\
\infty & \infty & 4 & \infty & 5 & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & 7 & \infty & \infty & \infty \\
\infty & \infty & \infty & \infty & 1 & \infty & 8 & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & 7 & \infty & 4 & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 8 \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & 6 & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 5 \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty
\end{pmatrix}
$$

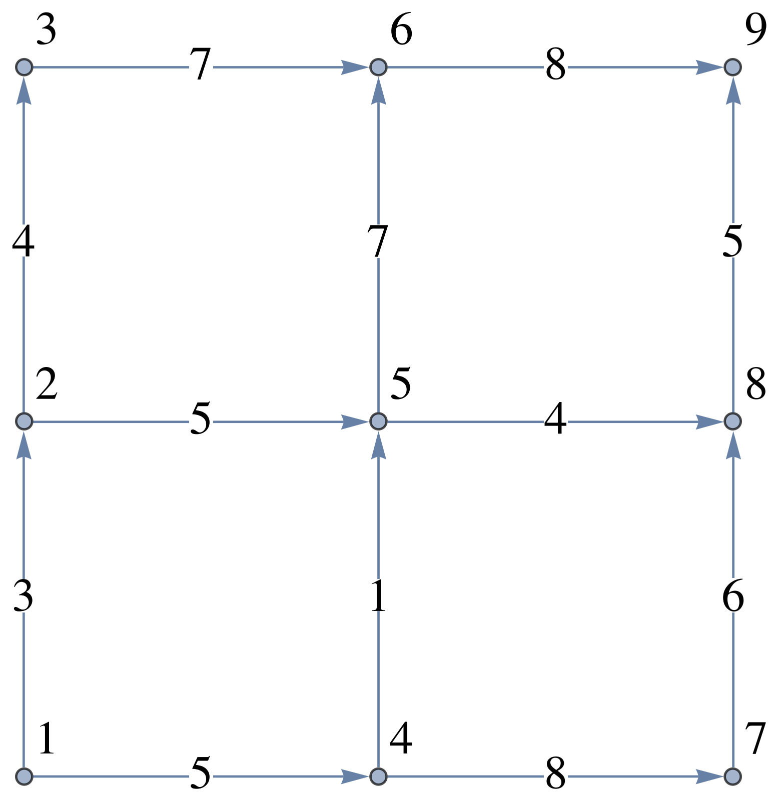


$$
\begin{aligned}
\text{edges} = \{&\{1 \to 2, 3\}, \{1 \to 4, 5\}, \{2 \to 3, 4\}, \{2 \to 5, 5\}, \{3 \to 6, 7\}, \{4 \to 5, 1\}, \\
&\{4 \to 7, 8\}, \{5 \to 6, 7\}, \{5 \to 8, 4\}, \{6 \to 9, 8\}, \{7 \to 8, 6\}, \{8 \to 9, 5\}\}; \\
c = \text{SparseArray}&[\{\#1[[1,1]], \#[[1,2]]\} \to \#[[2]]\&/@\text{edges}, \{9, 9\}, \infty];
\end{aligned}
$$

Dynamic Programming

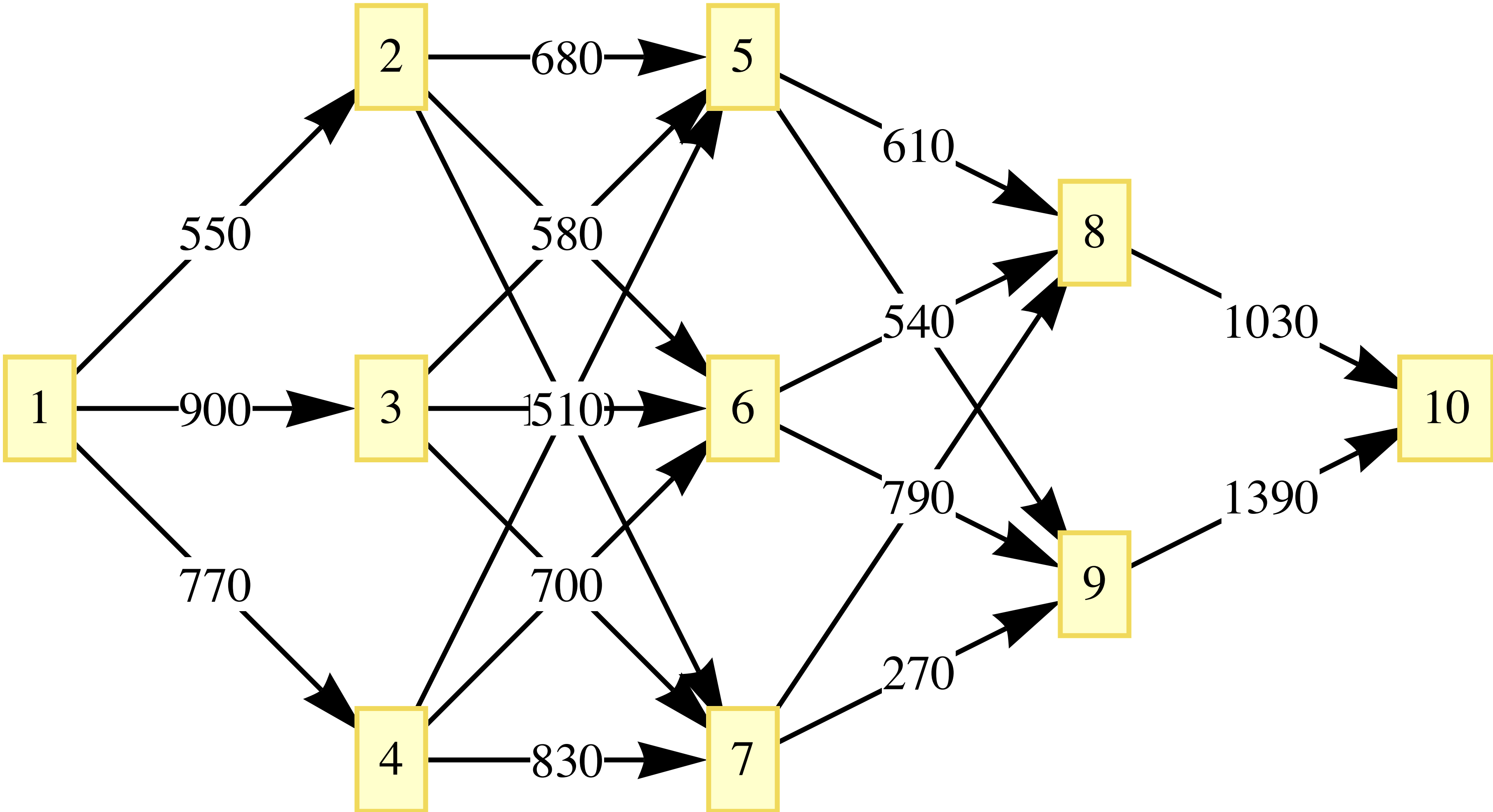
```
Clear[f, p]
p[T] = T;
f[T] = 0;
f[t_] := f[t] = Module[{v},
    v = Table[c[[t, j]] + f[j], {j, t + 1, T}];
    p[t] = Ordering[v, 1][[1]] + t;
    Min[v]]
```

Dynamic Programming



Voorbeeld #2 : Kortste pad

Dynamic Programming



Dynamic Programming

```
edges = {{1 -> 2, 550}, {1 -> 3, 900}, {1 -> 4, 770}, {2 -> 5, 680},
   {2 -> 6, 790}, {2 -> 7, 1050}, {3 -> 5, 580}, {3 -> 6, 760},
   {3 -> 7, 660}, {4 -> 5, 510}, {4 -> 6, 700}, {4 -> 7, 830},
   {5 -> 8, 610}, {5 -> 9, 790}, {6 -> 8, 540}, {6 -> 9, 940},
   {7 -> 8, 790}, {7 -> 9, 270}, {8 -> 10, 1030}, {9 -> 10, 1390}};
vertices = {1 -> {1, 1}, 2 -> {2, 2}, 3 -> {2, 1}, 4 -> {2, 0},
   5 -> {3, 2}, 6 -> {3, 1}, 7 -> {3, 0}, 8 -> {4, 1.5},
   9 -> {4, 0.5}, 10 -> {5, 1}};

T = 10;
c = SparseArray[{#1[[1, 1]], #[[1, 2]]} → #[[2]] & /@ edges,
   {T, T}, ∞];
c // MatrixForm
```

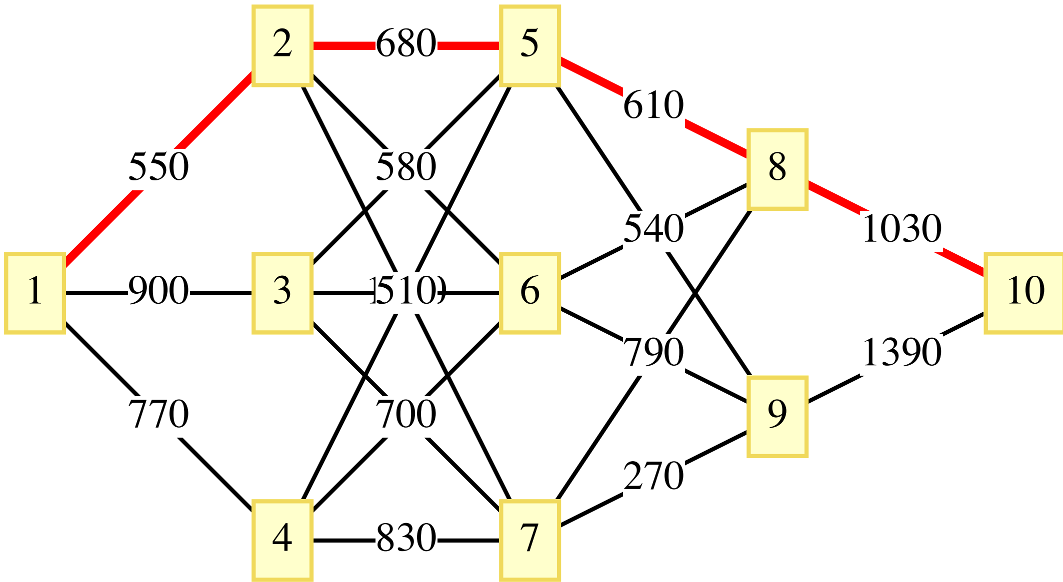
Dynamic Programming

```
T = 10;
c = SparseArray[{#1[[1, 1]], #[[1, 2]]} → #[[2]] & /@ edges,
    {T, T}, ∞];
c // MatrixForm
```

$$c = \begin{pmatrix} \infty & 550 & 900 & 770 & \infty & \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & 680 & 790 & 1050 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & 580 & 760 & 660 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & 510 & 700 & 830 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & 610 & 790 & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & 540 & 940 & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & 790 & 270 & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 1030 \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 1390 \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty \end{pmatrix}$$

Dynamic Programming

```
Clear[f, p]
p[T] = T;
f[T] = 0;
f[t_] := f[t] = Module[{v},
    v = Table[c[[t, j]] + f[j], {j, t + 1, T}];
    p[t] = Ordering[v, 1][[1]] + t;
    Min[v]]

f[1]

2870

y = 1;
path = Join[{1},
  Table[y = p[y], {t, 4}]]

{1, 2, 5, 8, 10}
```



Dynamic Programming

Voorbeeld #3 : Voorraad beheer

Dynamic Programming

- Voorraad beheer:
  - Planning periode : ***T***
  - In maand ***t*** is de vraag naar een product d_t en de hoeveelheid geproduceerde producten x_t
  - De voorraad aan het einde van maand t is i_t
  - De productie in maand ***t*** vervult de behoefte in maand ***t***
    - Neem aan dat in maand ***t*** de totale kosten gegeven worden door

$$c(x_t, i_t) = a + b\,x_t + h\,i_t \text{ if } x_t > 0$$

$$c(x_t, i_t) = h\,i_t \text{ if } x_t = 0.$$

    - Met de beperking dat

$$0 \le x_t \le r$$

$$0 \le i_t \le s.$$

  - Bereken de optimale x_t

```
a = 3; b = 1; h = 0.5;
r = 5; s = 4; T = 4;
d = {1, 3, 2, 4};
```

Dynamic Programming

- Als $f_t(i)$ de *minimale kosten* zijn om aan de behoefte van de maanden $t, t+1, \ldots, T$ te voldoen dan volgt dat

$$f_t(i) = \min_{x_t} \left[c(x_t, i + x_t - d_t) + f_{t+1}(i + x_t - d_t) \right]$$

  - (productie x_t, voorraad i, vraag d_t)

- De optimale waarde van x_i is $p_t(i)$

- In de laatste maand moet de voorraad leeg zijn

$$p_T(i) = d_T - i$$
$$f_T(i) = c(d_T - i, 0)$$

Dynamic Programming

```
Remove["Global`*"]

a = 3; b = 1; h = 0.5; r = 5; s = 4; T = 4;
d = {1, 3, 2, 4};

c[x_, i_] = Piecewise[{{0, x == 0}, {a + b x, x > 0}}, ∞] + h i
```

$$0.5\,i + \left(\begin{cases} 0 & x == 0 \\ 3 + x & x > 0 \\ \infty & \text{True} \end{cases} \right)$$

```
Clear[f, p]
p[T, i_] := d[[T]] - i;
f[T, i_] := f[T, i] = c[d[[T]] - i, 0]
f[t_, i_] := f[t, i] = Module[{v, e = d[[t]]},
    v = Table[c[x, i + x - e] + f[t + 1, i + x - e], {x, e - i, Min[r, s + e - i]}];
    p[t, i] = Ordering[v, 1][[1]] + e - i - 1;
    Min[v]]
```

Dynamic Programming

```
Grid[Join[{Join[{"Init. inv."}, Range[0, s]]},
   Table[Join[{Row[{"Month ", t}]}, Table[p[t, i], {i, 0, s}]], {t, T}]],
  Dividers → {2 → True, 2 → True}, Alignment → {Left, {Right}}, Spacings → 2,
  Background → {Automatic, Automatic,
    {{{2, 3}, {2, 2}} → LightGray, {{4, 4}, {4, 4}} → LightGray,
     {{5, 5}, {2, 2}} → LightGray}}]
```

Init. inv.	0	1	2	3	4
Month 1	1	0	0	0	0
Month 2	5	4	3	0	0
Month 3	2	5	0	0	0
Month 4	4	3	2	1	0

```
a = 3; b = 1; h = 0.5;
r = 5; s = 4; T = 4;
d = {1, 3, 2, 4};
```

- Maand 1: Beginvoorraand 0, (vraag maand 1 is 1 unit), produceer 1 unit
- Maand 2: Beginvoorraand 0, (vraag maand 1 was 1 unit), produceer 5 units
- Maand 3: Beginvoorraand 2, (vraag maand 2 was 3 units), produceer 0 units
- Maand 4: Beginvoorraand 2, (vraag maand 3 was 2 units), produceer 4 units

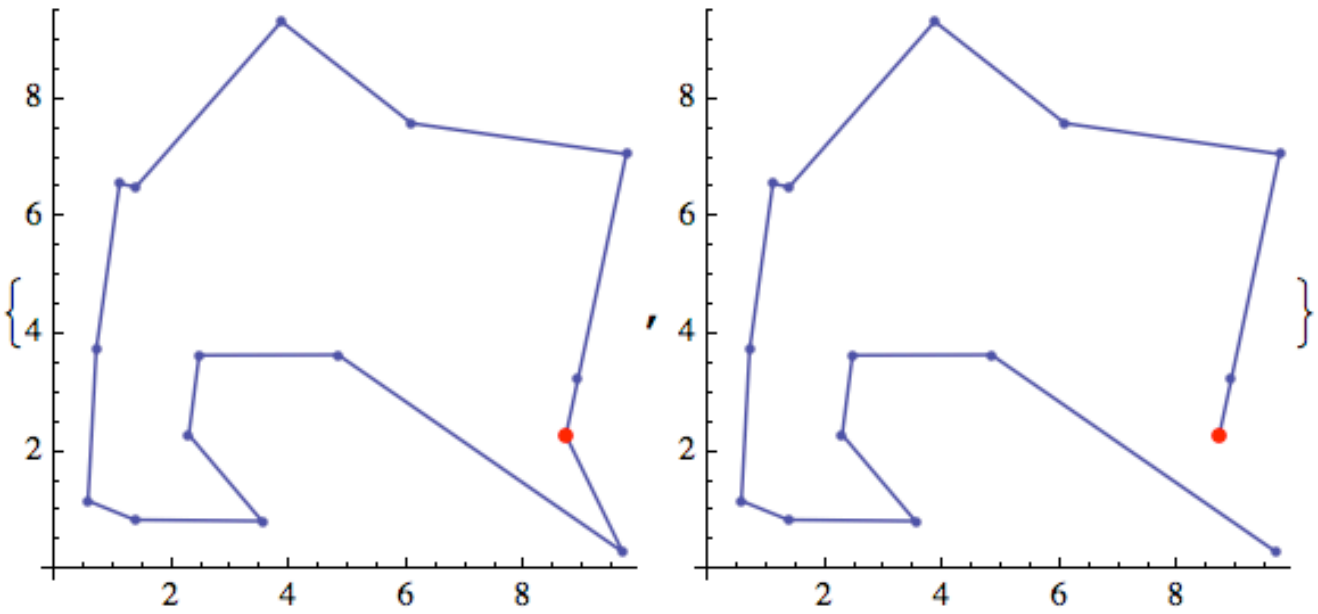
Voorbeeld #4 : Traveling salesman

Dynamic Programming

```
Remove["Global`*"]

T = 15; SeedRandom[54];
points = RandomReal[10, {T, 2}];
c = Array[cc, {T, T}];
Table[c[[i, j]] = EuclideanDistance[points[[i]], points[[j]]], {i, T}, {j, T}];

Clear[f, p]; cities = Range[2, T];
p[T, i_, cities] := 1
f[T, i_, cities] := f[T, i, cities] = c[[i, 1]]
f[t_, i_, S_] := f[t, i, S] = Module[{unvisited, v},
   unvisited = Complement[cities, S];
   v = c[[i, #]] + f[t + 1, #, Union[S, {#}]] & /@ unvisited;
   p[t, i, S] = Extract[unvisited, Ordering[v, 1][[1]]];
   Min[v]]

f[1, 1, {}] // Timing

{12.7861, 37.6344}

i = 1; S = {}; tour1 = Join[{1}, Table[y = p[t, i, S]; i = y; S = Union[S, {y}]; y, {t, T}]]

{1, 7, 13, 2, 14, 10, 5, 4, 12, 15, 3, 9, 11, 6, 8, 1}
```

Dynamic Programming

```
({length, tour2} = FindShortestTour[points]) // Timing
{0.073891, {37.6344, {1, 7, 13, 2, 14, 10, 5, 4, 12, 15, 3, 9, 11, 6, 8}}}

ListLinePlot[points[[#]], Mesh → All, AspectRatio → Automatic,
   Epilog → {Red, PointSize[Medium], Point[First[points]]}] & /@ {tour1, tour2}
```



Vragen?

- Jan van Hulzen : j.r.van.hulzen@hva.nl

- Liv Harkes : l.harkes@hva.nl