



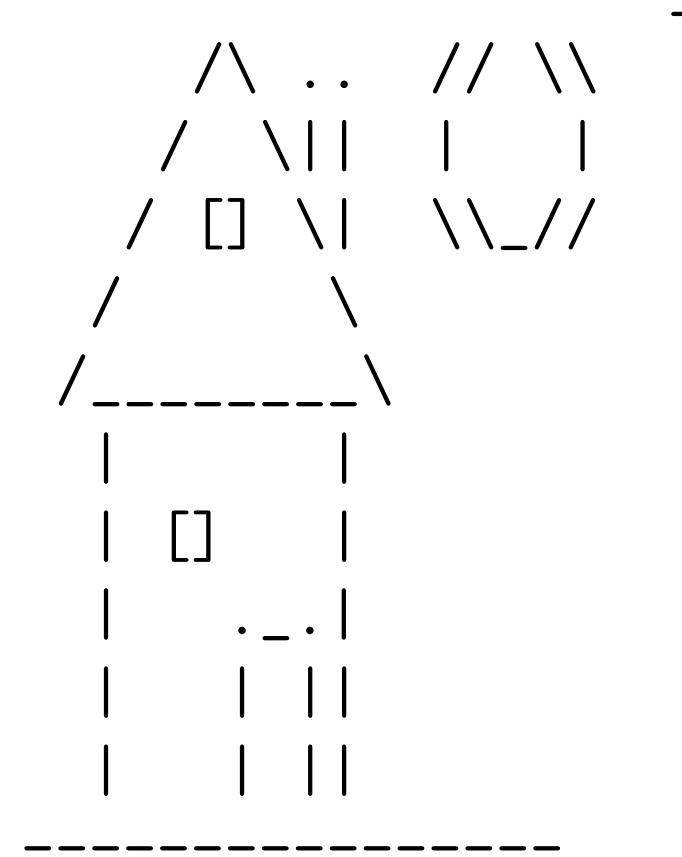
Perspektiverende Datalogikkursus

Uge 1 - Algoritmer og kompleksitet

Gerth Stølting Brodal

2. september 2005

Afleveringsopgaver...





Øvelse 10

Betrægt følgende liste af tal.

30 83 73 80 59 63 41 78 68 82 53 31 22 74 6 36 99 57 43 60

Øvelsen er at slette så få af disse tal som muligt, så de resterende tal står i voksende orden...

Indhold

- **Eksempler på beregningsproblemer**
- Algoritmer og deres analyse
 - Korrekthed af algoritmer
 - Ressourceforbrug for algoritmer
- Kompleksitet af beregningsproblemer

Beregningsproblemer

- Sortering
- Søgning
- Grafer
- Strenge
- Kombinatorisk optimering
- Geometri
- Numeriske beregninger
- :



Sortering

Problem: Stil en mængde elementer i orden.

```
110 755 766 51 652 28 729 713 681 407  
↓  
28 51 110 407 652 681 713 729 755 766
```

Data er meget bekvemmere hvis de er sorterede. Specielt er det nemmere at lede i dem (ordbøger, telefonbøger, eksamensopslag,...).

Brugt som rutine i mange andre algoritmer.

Meget velstuderet problem.

Mange algoritmer (jvf. øvelserne + QuickSort + ShellSort + ...).

Søgning

Problem: Gem data så de kan findes igen effektivt.

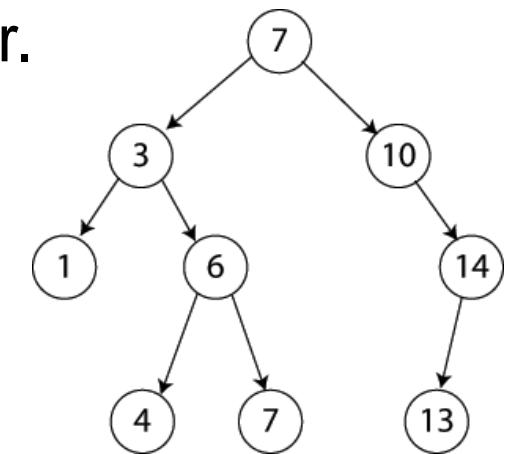
- $\text{Find}(x)$
- $\text{Insert}(x)$
- $\text{Delete}(x)$
- $\text{Successor}(x)$, $\text{RangeSearch}(x_1, x_2)$, ...

Et andet meget fundamentalt problem (jvf. databaser).

Brugt som rutine i mange andre algoritmer.

Meget velstuderet, mange algoritmer.

To grundgrupper: søgetræer og hashing.



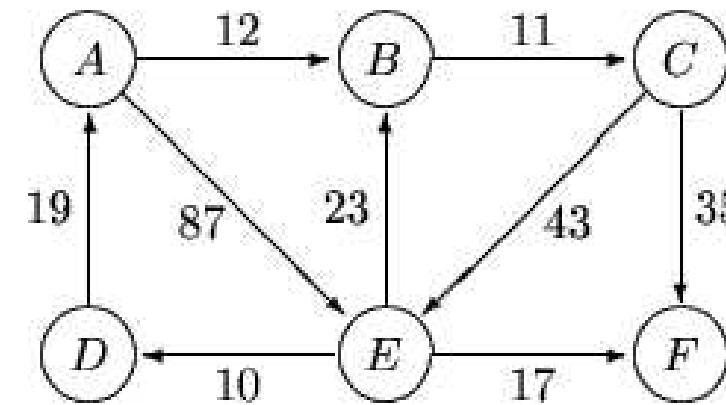
Grafer

Knuder (punkter) og kanter (stregen mellem punkter).

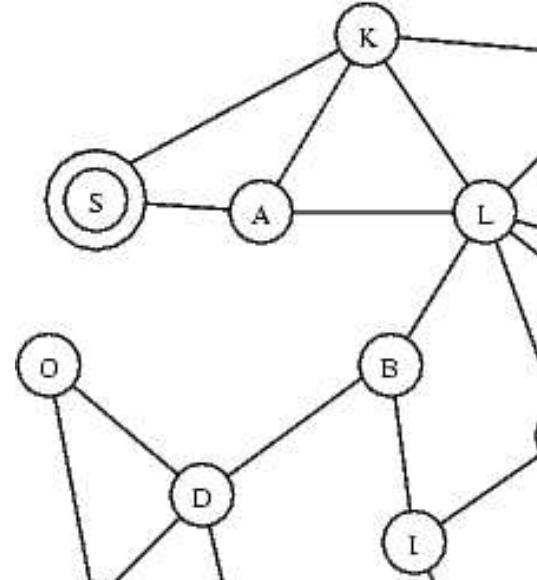
Ekstra struktur: orientering af kanter, vægte på kanter.

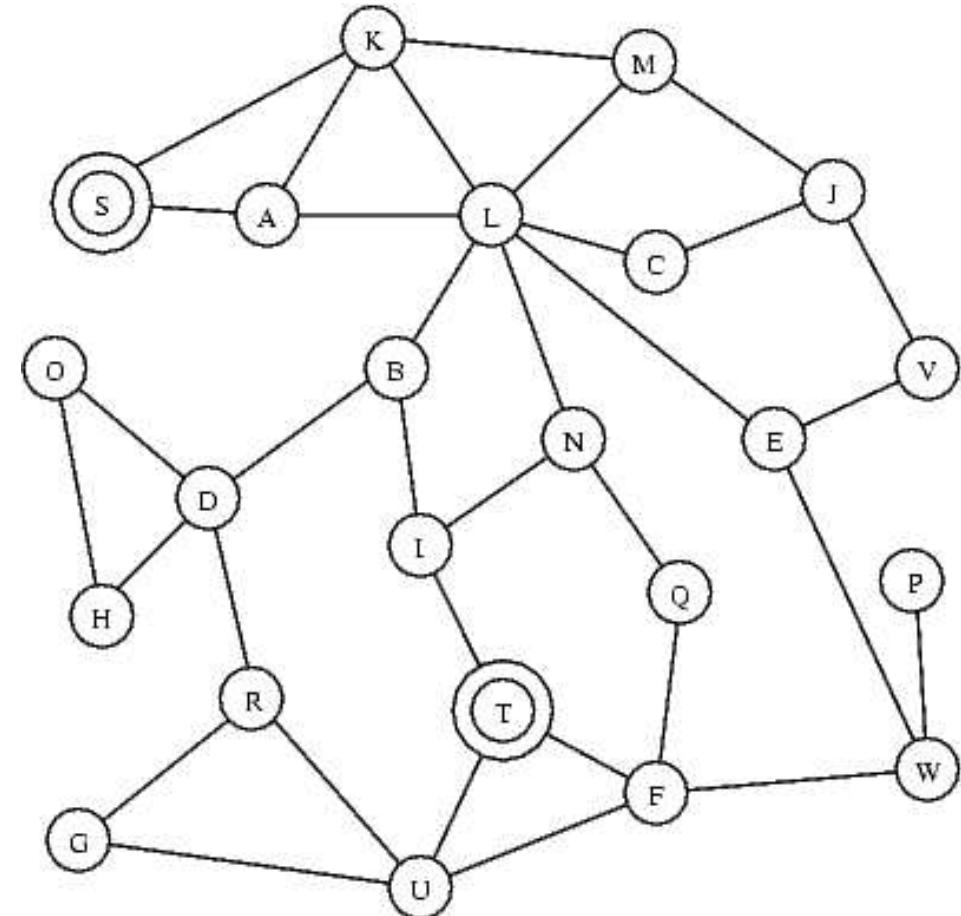
En *meget* anvendelig model:

Flyruter, veje, el/vand/computer netværk,
bekendtskaber og andre relationer,...



Problemer på grafer

- Løb grafen igennem (besøge alle knuder).
 - Sammenhæng, k -sammenhæng.
 - (Mindste) udspændende træ.
 - Korteste veje.
 - Euler tur.
 - Hamilton tur,
rejsende sælger.
 - Graffarvning.
 - Klike





Strenge

Alfabet = mængde af tegn som kan bruges

Streng = sekvens af tegn fra alfabetet

Eksempler:

“To be or not to be”

ACCCATTCCGTAA

10001100110001110

Problemer på strenge

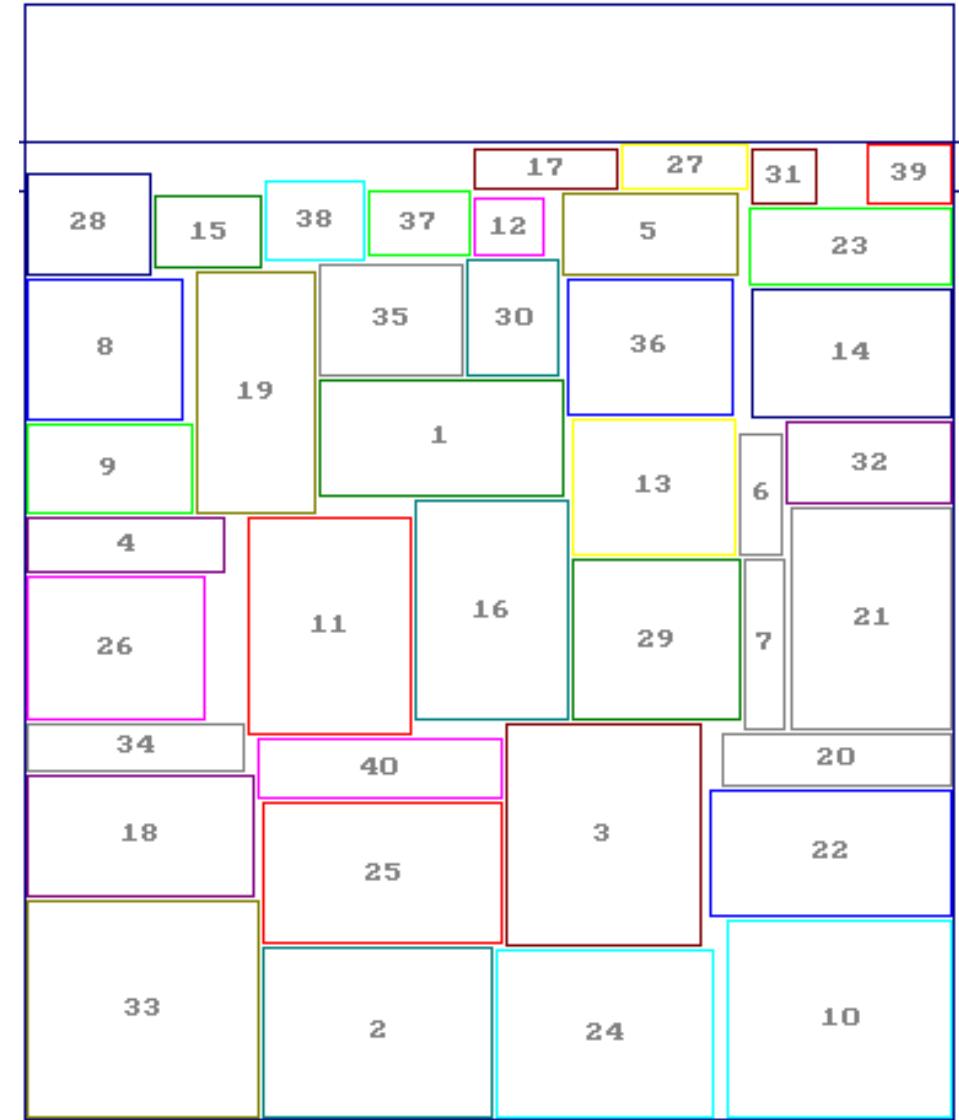
- Mønster genkendelse
- Regulære udtryk
- Afstandsmål (Hamming, edit)
- Længste fælles delstrenge
- Længste fælles delsekvens
- :



The diagram shows two DNA sequences aligned vertically. The top sequence is: ACGTTACGTAAACTACTAGTACACACACTACCCAT. The bottom sequence is: ACAGTGTTAAGTAAACTAACTAGTACACACGTACCCAT. Red diagonal lines connect corresponding bases between the two sequences, indicating matches. There are several mismatches and gaps represented by dashes ('-') in the bottom sequence. The first mismatch is at position 1 (A vs C), and the second is at position 2 (G vs T). There is a gap of three bases starting at position 4 (AAG). The third mismatch is at position 7 (A vs G). The fourth mismatch is at position 10 (C vs T). The fifth mismatch is at position 12 (A vs C). The sixth mismatch is at position 14 (A vs C). The seventh mismatch is at position 16 (A vs G). The eighth mismatch is at position 18 (C vs T). The ninth mismatch is at position 20 (A vs C). The tenth mismatch is at position 22 (A vs G). The eleventh mismatch is at position 24 (C vs T). The twelfth mismatch is at position 26 (A vs C). The thirteenth mismatch is at position 28 (A vs G). The fourteenth mismatch is at position 30 (C vs T). The fifteenth mismatch is at position 32 (A vs C). The sixteenth mismatch is at position 34 (A vs G). The seventeenth mismatch is at position 36 (C vs T). The eighteenth mismatch is at position 38 (A vs C). The nineteenth mismatch is at position 40 (A vs G). The twentieth mismatch is at position 42 (C vs T). The twenty-first mismatch is at position 44 (A vs C). The twenty-second mismatch is at position 46 (A vs G). The twenty-third mismatch is at position 48 (C vs T). The twenty-fourth mismatch is at position 50 (A vs C). The twenty-fifth mismatch is at position 52 (A vs G). The twenty-sixth mismatch is at position 54 (C vs T). The twenty-seventh mismatch is at position 56 (A vs C). The twenty-eighth mismatch is at position 58 (A vs G). The twenty-ninth mismatch is at position 60 (C vs T). The thirtieth mismatch is at position 62 (A vs C). The thirtieth mismatch is at position 64 (A vs G). The thirtieth mismatch is at position 66 (C vs T). The thirtieth mismatch is at position 68 (A vs C). The thirtieth mismatch is at position 70 (A vs G). The thirtieth mismatch is at position 72 (C vs T). The thirtieth mismatch is at position 74 (A vs C). The thirtieth mismatch is at position 76 (A vs G). The thirtieth mismatch is at position 78 (C vs T). The thirtieth mismatch is at position 80 (A vs C). The thirtieth mismatch is at position 82 (A vs G). The thirtieth mismatch is at position 84 (C vs T). The thirtieth mismatch is at position 86 (A vs C). The thirtieth mismatch is at position 88 (A vs G). The thirtieth mismatch is at position 90 (C vs T). The thirtieth mismatch is at position 92 (A vs C). The thirtieth mismatch is at position 94 (A vs G). The thirtieth mismatch is at position 96 (C vs T). The thirtieth mismatch is at position 98 (A vs C). The thirtieth mismatch is at position 100 (A vs G).

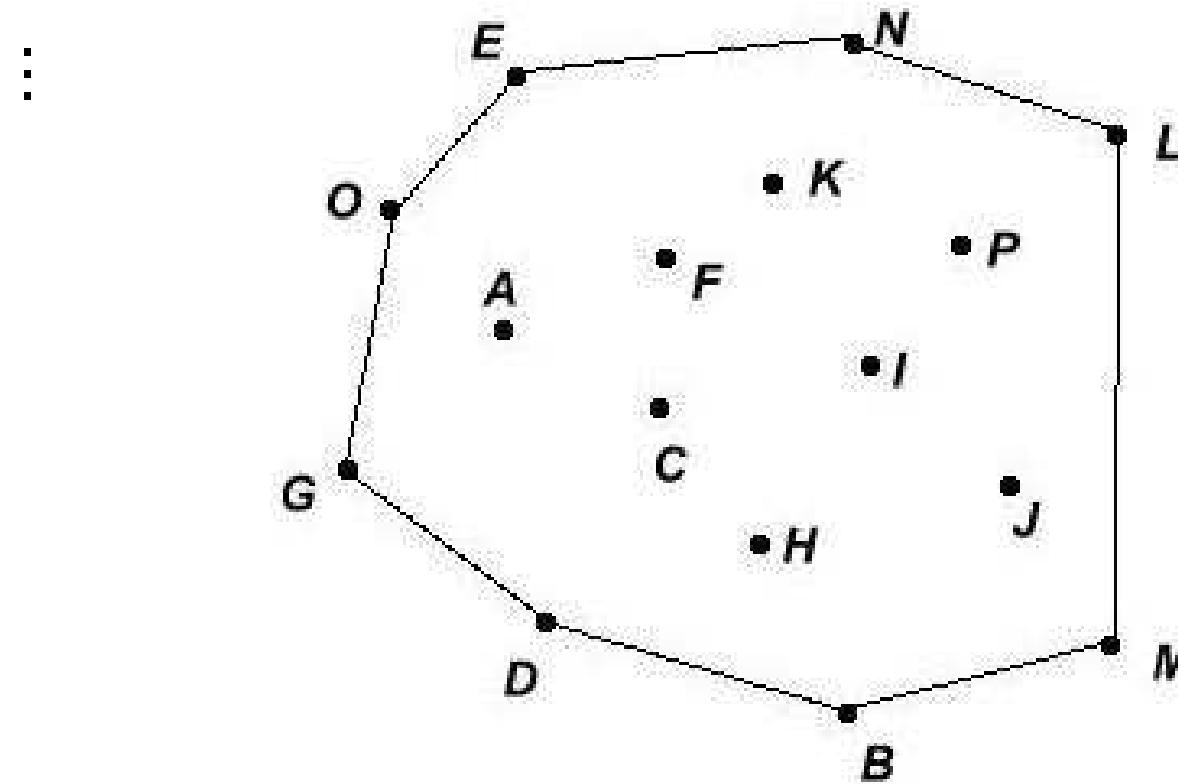
Kombinatorisk optimering

- Bin Packing (1D, 2D, 3D)
- Knapsack
- Subset Sum
- Job Scheduling
- Crew assignment
- ...



Geometri

- Convex Hull
- Nearest Neighbor
- Orthogonal Line Segment Intersection
- 2D Range Search



Numeriske beregninger

- Polynomieevaluering
- Matrixmultiplikation
- Løsning af ligningssystemer
- Løsning af differentialligninger

⋮

$$(6/7-1)*7+1 = -4.44089209850063e-16 ?$$



Indhold

- Eksempler på beregningsproblemer
- Algoritmer og deres analyse
 - **Korrekthed af algoritmer**
 - Ressourceforbrug for algoritmer
- Kompleksitet af beregningsproblemer

Invarianter

Udsagn I som gælder efter alle skridt i algoritmen.

Vælges så:

- Man kan vise at I gælder ved starten.
- Man kan vise at hvis I gælder før et skridt, så gælder det efter.
- Man kan vise af I samt omstændigheder ved algoritmens afslutning implicerer det ønskede slutresultat.

Eksempler: RadixSort, binær søgning.

Indhold

- Eksempler på beregningsproblemer
- Algoritmer og deres analyse
 - Korrekthed af algoritmer
 - **Ressourceforbrug for algoritmer**
- Kompleksitet af beregningsproblemer

Ressourceforbrug, målestok

- RAM-modellen.
- Tidsmåling vs. analyse.
- Voksehastighed, asymptotisk notation.
- Worst case, best case, average case.

Først vælge (eller udvikle) algoritme efter forskelle i asymptotisk ressourceforbrug

Ved lighed, vælg dernæst efter konstanter (tidsmåling nu relevant).

Indhold

- Eksempler på beregningsproblemer
- Algoritmer og deres analyse
 - Korrekthed af algoritmer
 - Ressourceforbrug for algoritmer
- **Kompleksitet af beregningsproblemer**

Nedre grænser

Beviser for at **ingen** algoritme (blandt en stor klasse af algoritmer for en given beregningsmodel) kan løse problemet bedre end angivet.

Eksempler:

- Sortering
- Søgning

Øvre og nedre grænser ens
↓
problemets kompleksitet kendt.



$P \subseteq EXP$

Meget grov inddeling af algoritmer i gode og dårlige:

P = problemer med **polynomiel tids** algoritme
v.s.

EXP = problemer med **eksponentiel tids** algoritme

Eksempel: sortering vs. brute-force løsning af puslespil



NP

NP = ja/nej-problemer, hvor en ja-løsning kan kontrolleres
(men ikke nødvendigvis findes) i polynomiel tid.

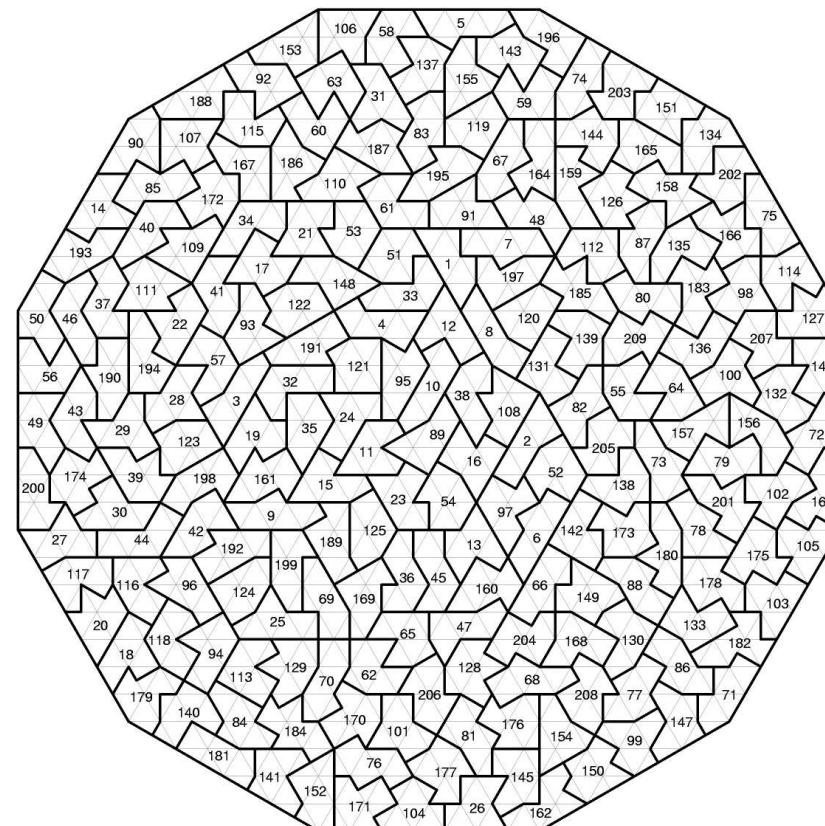
Eksempel: Hamilton tur.

Det er nemt at se at $P \subseteq NP \subseteq EXP$.

Formodning: $P \subsetneq NP$

Hvis ingen polynomiel algoritme..

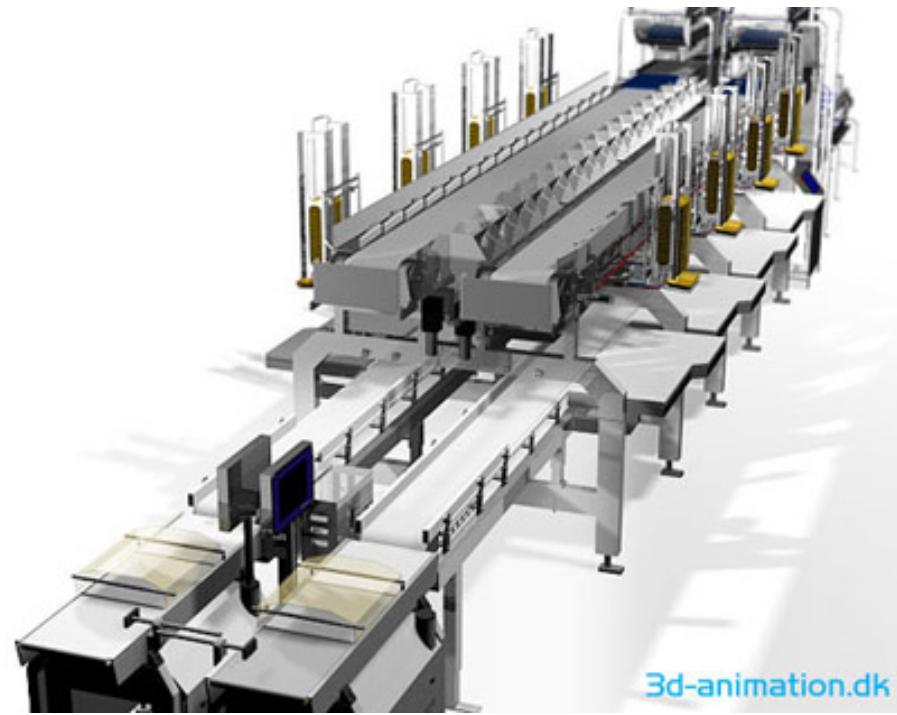
- Heuristisk søgning
- Algoritmer for specielle instanser
(jvf. “The Eternity Puzzle”).
- Approximationsalgoritmer



Flere modeller og cost-funktioner

- Online algoritmer
- Randomiserede algoritmer
- Parallelisme
- Hukommelseshierarkier

:



Algoritmer og kompleksitet

- Udvikle algoritmer
- Analysere algoritmer
- Analysere problemer

David Harel:

*“Algorithmics is more than a branch of computer science. It is the **core of computer science**, and, in all fairness, can be said to be relevant to most of science, business, and technology.”*

