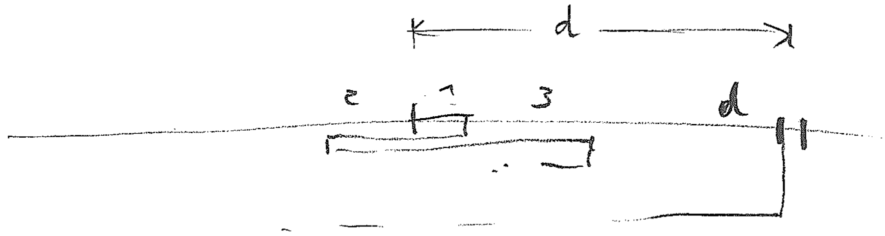
On-line-Algorithmen

0. Org

1. Einführung — Tür in der Wand

- müssen beide Seiten erkunden
- Vergrößerung der Suchtiefe?
  - jeweils um 1 Einheit (Schritt, Meter)



3 formulae

gelaufener Weg: $2(1+2+\dots+(d+1))+d = 2\cdot\frac{(d+1)(d+2)}{2}+$

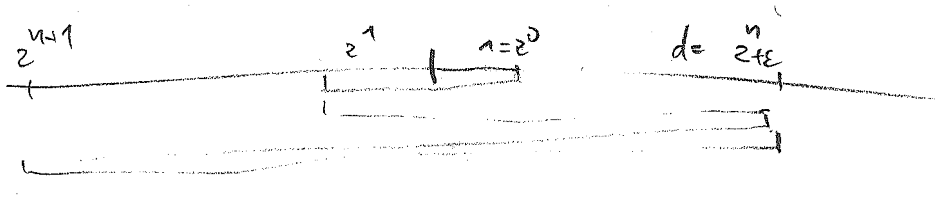
$= d^2+4d$

kürzester Weg: d

(falls $d \approx 100$ m: $d^2 \approx 10$ km)

Geht's besser?

- jeweils verdoppeln; worst case: Tür knapp verfehlt:

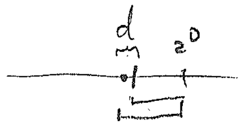


gelaufener Weg: $2(1+2+2^2+\dots+2^n+2^{n+1})+2^n+\varepsilon = 2\,\frac{2^{n+2}-1}{2-1}+$

$= 8\cdot 2^n - 2 + 2^n+\varepsilon$

$< 9\cdot(2^n+\varepsilon) = 9\cdot d$

Tür links vom Start: analog.

Tür im Abstand $d < 2^1$ links: 

gelaufener Weg $= 2(2^0) + d < 9d + 2$.

Damit: Bei Verwendung der Strategie "Tiefenverdopplung" gilt:

$$\text{gelaufener Weg} \leq 9 \cdot \text{kürzester Weg} + 2 .$$

Solche Aussagen sind Ziel dieser Vorlesung!

Allgemeine Definition:

Sei Π ein Problem (Tür in der Wand finden)

$P \in \Pi$ eine Instanz von Π (konkrete Wand mit konkreter Tür)

$OPT_\Pi(P)$ eine optimale Lösung der Instanz P (gehe direkt Tür)

S eine Strategie zur Lösung von Π (z.B. Tiefenverdopp

Wir sagen: S ist c-kompetitive Lösung von Π,

falls gilt: es gibt $A > 0$ mit

$$\boxed{\forall P \in \Pi : \quad S(P) \leq c \cdot OPT_{\Pi}(P) + A}$$

Falls $A \leq 0$: S ist stark c-kompetitiv.

Strategie $\hat{=}$ Algorithmus

Typisch: S kennt Π (die Menge der möglichen Inputs)

S lernt P erst zur Laufzeit kennen.

Proposition 1.1 Tiefenverdopplung ist 9-kompetitive Strategie für das Problem "Tür in der Wand finden".

Allgemein: Gegeben Problem Π.

Interessante Fragen:

- Gibt es eine c-kompetitive Lösung S für Π?

  Wenn ja, was ist das kleinste c, für das dieses S c-kompetitiv ist?

- Was ist das kleinste c, für das eine c-kompetitive L von Π existiert?

  $\inf \{ c \geq 1 ;$ ex. c-kompetitive Lösung für $\Pi \}$

  heißt die kompetitive Komplexität von Π.

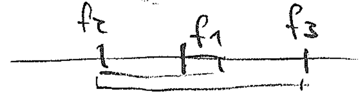
(leider nur für wenige Probleme bekannt)

Theorem 12 (Gal, Baeza-Yates et al., ..)

Das Problem "Tür in der Wand finden" hat die kompetitive Komplexität 9.

Beweis Sei S eine c-kompetitive Lösung. (Angenommen: c <

S gegeben durch Folge $(f_1, f_2, f_3, \dots)$



Muß gelten: $\forall n : \forall \varepsilon :$ (falls Tür in Entfernung $f_n + \varepsilon$)

$$2(f_1 + \dots + f_n + f_{n+1}) + f_n + \varepsilon \leq c(f_n + \varepsilon) + A$$

$$\Rightarrow \quad 2(f_1 + \dots + f_n + f_{n+1}) + f_n \leq c f_n + A$$

nehmen zunächst an: $A = 0$

$$\Rightarrow \quad f_1 + \dots + f_{n-1} + f_{n+1} \leq \underbrace{\frac{c-3}{2}}_{=: H \, < 3} f_n$$

$$\Rightarrow \quad f_{n+1} \le H f_n - \sum_{i=1}^{n-1} f_i \qquad \forall n \ge 1$$

einsetzen: $f_n \le H f_{n-1} - \sum_{i=1}^{n-2} f_i$

$$\le H^2 f_{n-1} - H \sum_{i=1}^{n-2} f_i - \sum_{i=1}^{n-1} f_i$$

$$= (H^2-1) f_{n-1} - (H+1) \sum_{i=1}^{n-2} f_i \qquad \forall n \ge 2$$

einsetzen ...

$$(*) \qquad \le a_m f_{n-m} - b_m \sum_{i=1}^{n-1-m} f_i \qquad \forall n \ge 1, \ \forall\, 0 \le m \le n-1$$

wobei die Koeffizienten a_m, b_m definiert sind durch

$$a_0 := H \quad ; \quad b_0 := 1$$

simultane lineare Rekursion

$$a_{i+1} := a_i H - b_i \quad ; \quad b_{i+1} := a_i + b_i$$

genau das passiert beim Einsetzen

(Beweis durch Ind. (i)).

<u>Trick</u> (zur Lösung von simultanen linearen Rekursionen):

<u>1. Darstellung als Matrixmultiplikation</u>

$$\begin{pmatrix} a_{i+1} \\ b_{i+1} \end{pmatrix} = \underbrace{\begin{pmatrix} H & -1 \\ 1 & 1 \end{pmatrix}}_{=:M} \begin{pmatrix} a_i \\ b_i \end{pmatrix} = M^{i+1} \begin{pmatrix} a_0 \\ b_0 \end{pmatrix}.$$

Idea: express $\begin{pmatrix} a_0 \\ b_0 \end{pmatrix}$ as linear combination of Eigenvectors!

2. Berechnung der Eigenvektoren von M

Eigenwerte $z, \bar{z}$: Nst. des char. Polynoms $\chi_M(t) = |tE - M|$

$$= \begin{vmatrix} t-H & 1 \\ -1 & t-1 \end{vmatrix}$$

$$= t^2 - (H+1)t + H+1$$

$(H+1)^2 - 4\cdot 1\cdot(H+1) = (H-3)(H+1)$

$$\Rightarrow z, \bar{z} = \frac{1}{2}\left(H+1 \pm \sqrt{\underbrace{(H-3)(H+1)}_{<0 \text{ wegen } H<3}} \right)$$

<0 wegen $H<3$, also $z, \bar{z}$ konjugiert komplex, $z \in \mathbb{C} \setminus \mathbb{R}$.

Eigenvektoren $V, \bar{V}$:

Lösungen des linearen Gleichungssystems $M \begin{pmatrix} x \\ y \end{pmatrix} = z \begin{pmatrix} x \\ y \end{pmatrix}$

$$V = \begin{pmatrix} 1 \\ \bar{z}-1 \end{pmatrix}, \qquad \bar{V} = \begin{pmatrix} 1 \\ z-1 \end{pmatrix}$$

linear unabhängig, also Basis vom $\mathbb{R}$

3. Damit Lösung der Rekursion

$$\begin{pmatrix} a_0 \\ b_0 \end{pmatrix} = v V + \bar{v} \bar{V} \qquad \text{mit } v \in \mathbb{C}$$

bestimmen

$$\Rightarrow \begin{pmatrix} a_1 \\ b_1 \end{pmatrix} = M \begin{pmatrix} a_0 \\ b_0 \end{pmatrix} = v M V + \bar{v} M \bar{V}$$

$$= v z V + \bar{v} \bar{z} \bar{V}$$

($V, \bar{V}$ Eigenvektoren von M)

$\vdots$

$$\Rightarrow \begin{pmatrix} a_{i+1} \\ b_{i+1} \end{pmatrix} = v z^{i+1} V + \bar{v} \bar{z}^{i+1} \bar{V}$$

$$\Rightarrow a_{i+1} = v z^{i+1} + \bar{v} \bar{z}^{i+1} = 2 \operatorname{Re}(v z^{i+1})$$

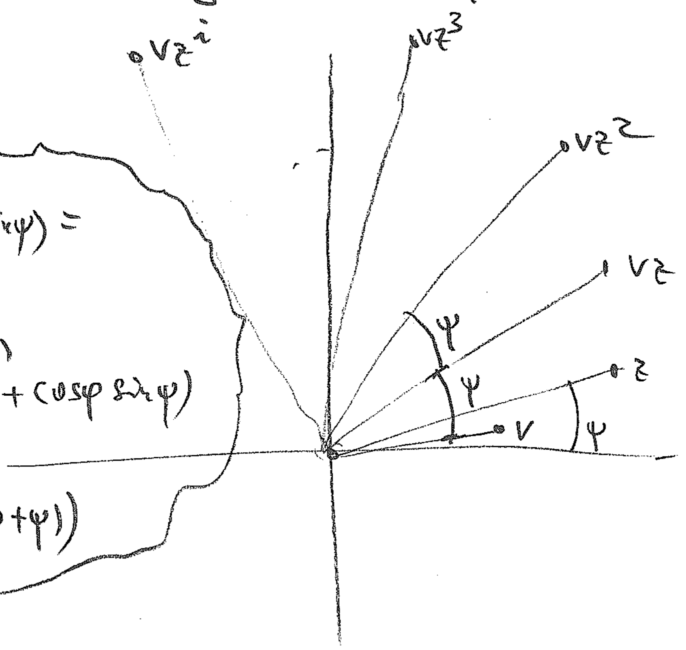
Realteil

$$\left(a+ib \in \mathbb{C} \Rightarrow a+ib + \overline{(a+ib)} = a+ib + a-ib = 2a \right).$$

Geometrische Deutung der Multiplikation mit z:

Drehung um festen Winkel + Streckung

(Add.-Thm. sin cos)



$(\cos\varphi, i\sin\varphi)(\cos\psi, i\sin\psi) =$

$(\cos\varphi\cos\psi - \sin\varphi\sin\psi,$

$\quad i(\sin\varphi\cos\psi + \cos\varphi\sin\psi))$

$= (\cos(\varphi+\psi), i\sin(\varphi+\psi))$

$\Rightarrow$ es gibt ein i : vz^i liegt in linker Halbebene

$\Rightarrow \quad \mathrm{Re}(vz^i) < 0$

$\Rightarrow \quad a_i < 0$

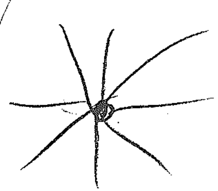
Das kann aber nicht sein! Denn

$$0 < f_{i+2} \underset{\substack{\uparrow \\ (*) \\ n=i+1 \\ m=i}}{\leq} a_i f_1 - b_i \cdot 0 = a_i \underbrace{f_1}_{>0} \Rightarrow a_i > 0.$$

$\longrightarrow$ (5.1

<u>Fragen</u>:

- Hilft Randomisierung weiter? $\leftarrow$
- Suche auf $m > 2$ Halbgeraden?



m

$\rightarrow$ Später (Navigation)

Kao/Reif/Tate 93

Let $\gamma \approx 3.591$ solution

$\gamma \ln \gamma =$

ratio $\gamma+1 \approx 4.591$ optimal $\Leftarrow$ {
pick left/right at random
pick $x \in [0,1)$ at random
l-th turning point $:= \gamma^x \cdot \gamma^l$

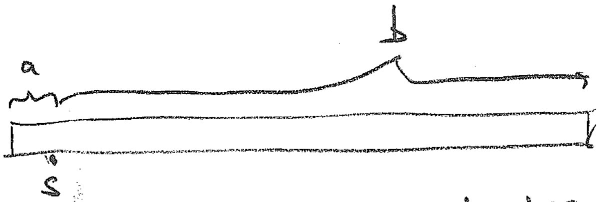
Müssen uns von additiver Konstante A noch befreien! Sei $\mu > 0$ beliebig.

Da $f_i \uparrow \infty$, gibt es n_0 mit $C f_n + A \le (C+\mu) f_n$

$$\Rightarrow \quad 2\sum_{i=n_0}^{n+1} f_i + f_n \le C f_n + A \le (C+\mu) f_n$$

analog weiter für $C+\mu$ statt μ

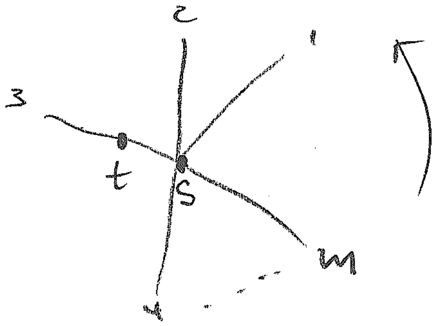
$$\Rightarrow \quad C+\mu \ge 9 \quad \underset{\mu>0 \text{ bel.}}{\Longrightarrow} \quad C \ge 9. \qquad \square$$



if $\frac{b}{a} > 9$, doubling beats dumb in the worst case

Also known as cow-path problem

instead of 2 half-lines: m



fix order

explore cyclically to depth of $f_j := \left(\frac{m}{m-1}\right)^j$

(exp(1); Euler numb

<u>Theorem</u> This strategy is $2\frac{m^m}{(m-1)^{m-1}} + 1 \le 2em+1$ —
competitive, (and this is optimal.)

Only a constant factor if m is a constant

<u>Proof of upper bound</u>

Suppose t lies at distance $f_j + \varepsilon$ from s

$\Rightarrow$ robot performs useless round of explorations of depth $f_{j+1}, \dots, f_{j+m-1}$ before returning to the correct halfline

$$\Rightarrow |path| \le 2\sum_{i=1}^{j+m-1} f_i + f_j + \varepsilon = 2\sum_{i=1}^{j+m-1}\left(\frac{m}{m-1}\right)^i + f_j + \varepsilon$$

$$= 2\,\frac{\left(\frac{m}{m-1}\right)^{j+m} - 1}{\frac{m}{m-1} - 1} + f_j + \varepsilon \qquad | \cdot (m-1)$$

$$< 2(m-1)\left(\frac{m}{m-1}\right)^{j+m} + f_j + \varepsilon$$

$$\Rightarrow \text{competitive factor} \le \frac{|path|}{f_j+\varepsilon} < \frac{2(m-1)\left(\frac{m}{m-1}\right)^{j+m}}{\left(\frac{m}{m-1}\right)^j} + 1$$

$$= \underbrace{2m\left(\frac{m}{m-1}\right)^{m-1} + 1}_{=9 \text{ if } m=2} = 2m \underbrace{\left(1-\frac{1}{m-1}\right)^{m-1}}_{\le e} + 1 \qquad \boxed{TM}$$

also known as round-robin doubling

doubling is more than a smart idea: a paradigm !

m processes, only one can be successful, don't know which one

1 processor

can stop any process at any time, but need to start from scratch when resuming.

different setting

m processors, every one will be successful, don't k[now] when

1 processor

can halt / continue any process at any time at no extra cost

→ hyperbolic dovetailing (Kirkpatrick '09)

Given: m lists of lengths $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_m \geq 1$ not in sorted order; lengths λ_i are unknown

Goal: traverse lists (using pointers) until end of at least one list is reached

(escape fro[m] a mine after ...)

(tu... hav... co...)

Theorem A strategy A knowing (the value set $\{\lambda_1, \dots, \lambda_m\}$) can finish in

$$W := \min_{1 \leq i \leq m} i \lambda_i$$

many steps, and not less.

Proof: Let A be such an algorithm.

Assume list i has been explored to depth d_i

$\hat{d}_i := i$-th largest of $\{d_1, \dots, d_m\}$

as long as $\hat{d}_1 < \lambda_1, \hat{d}_2 < \lambda_2, \dots, \hat{d}_m < \lambda_m$:

A is not done

if $\hat{d}_i \geq \lambda_i$ for some i :

because of $\hat{d}_1 \geq \hat{d}_2 \geq \dots \geq \hat{d}_i$:

i lists explored to depth $\geq \lambda_i$

$\Rightarrow$ A makes $\geq i \lambda_i \geq w$ steps.

Knowing $\{\lambda_1, \dots, \lambda_m\}$, we can compute w and that index j such that $w = j \lambda_j$.

Define strategy A by

explore all lists to depth λ_j, in any order

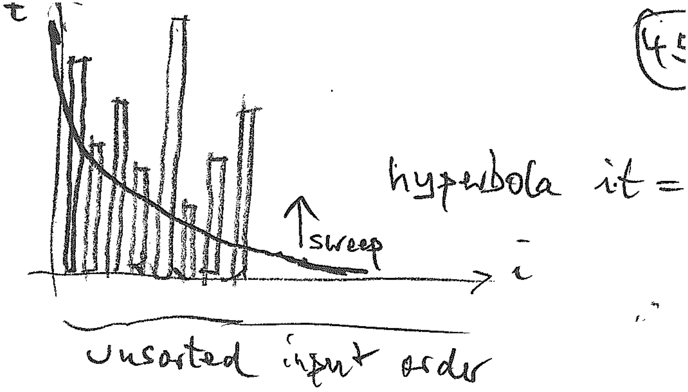
$\Rightarrow$ A terminates after $\leq j \lambda_j = w$ explorations;

because after $(j-1)\lambda_j$ useless exploration steps in the $j-1$ longest lists, A must exhaust a list of length

$\boxed{\text{Theo}}$

__Question__ How well can a strategy perform that doesn't know $\{\lambda_1, \dots, \lambda_m\}$?

Hyperbolic-Traversal



$c := 1$;
while no list fully explored do
  for $i := 1$ to m do
    Continue exploration to depth $\left\lfloor \frac{c}{i} \right\rfloor$;
  $c := c+1$

Theorem This strategy needs $(w+1)(\ln(\min(m,w))+1)$ steps

Proof Suppose Hyperbolic-Traversal has just performed a round of traversals for some c without success

Claim 1 $c \leq w$

Proof by ~~definition~~ of w, we need to show

$$c \leq i\, l_i \quad \forall i \quad \text{(sorted order)}$$

Let L be a list; by assumption, (otherwise, algorithm would have terminated)

$$c \leq \text{inputpos}(L) \cdot \text{length}(L)$$

if $\text{inputpos}(L) \leq \text{sortpos}(L)$: ✓

otherwise, $\text{sortpos}(L) < \text{inputpos}$.

Clear: there must be list L' such that

$$\text{inputpos}(L') \leq \text{sortpos}(L)$$
$$\text{length}(L') \leq \text{length}(L)$$

(not all lists at input positions $1, 2, \ldots, \text{sortpos}(L)$ can be longer than L)

$\Rightarrow$

$$c \leq \text{inputpos}(L') \cdot \text{length}(L') \leq \text{sortpos}(L) \cdot \text{length}(L).$$

$\boxed{\text{I}}$

all explored positions satisfy $i \cdot t \le c$ (used in last successful round)

at input position i, there are $\le \left\lfloor \frac{c}{i} \right\rfloor$ many

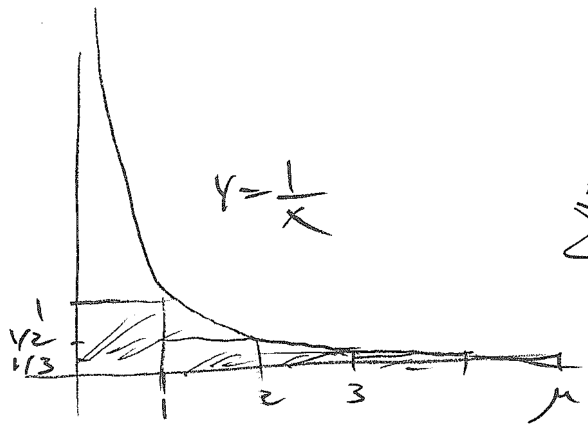
$\Rightarrow$ total number of positions explored

$$\le \sum_{i=1}^{m} \left\lfloor \frac{c}{i} \right\rfloor \le \sum_{i=1}^{\min(m,c)} \frac{c}{i} \le c\left(1 + \int_1^{\min(m,c)} \frac{1}{x}\,dx\right)$$

$$\le c\left(1 + \ln\left(\min(m,c)\right)\right)$$

$$\le (w+1)\left(1 + \ln\left(\min(m,w)\right)\right)$$

Claim 1



$$\sum^{\mu} \frac{1}{i} \le 1 + \int_1^{\mu} \frac{1}{x}\,dx = 1 + \ln \mu.$$

$\boxed{\text{Thm}}$

(formulae to memorize...)