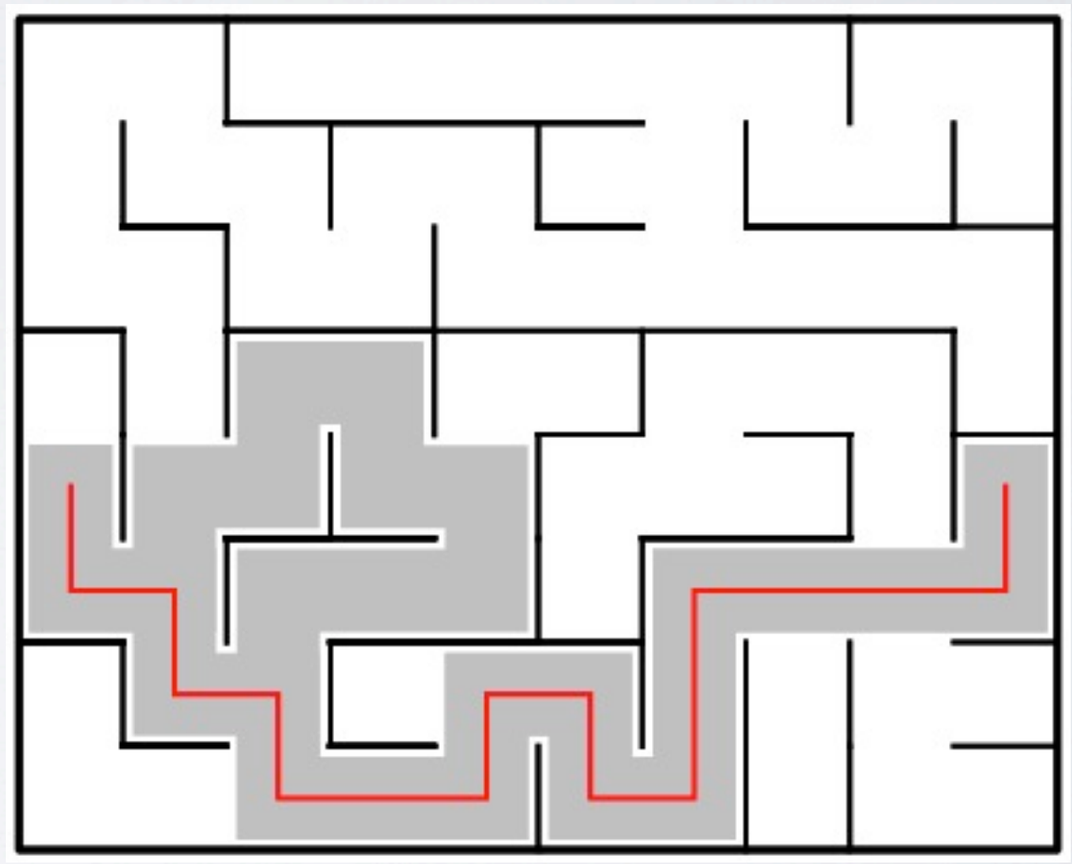
LABYRINTH SOLVER

Michael Borkowski
Alexander Falb

LABYRINTH SOLVER

- Input
  - Labyrinthbeschreibung
  - Start, Ziel
- Output
  - Alle Wege
  - "Bester" Weg



SPRACHSPEZIFISCHES

- Einlesen der Labyrinthbeschreibung mittels `cvx` und `exec`

```
/parseline { [ exch cvx exec ] } def

/getmatrix [
  (labyrinthfile) (r) file
  {
    dup 32 string readline
    not { pop exit } if

    dup length 0 ne
    { parseline exch }
    { pop } ifelse
  } loop
] def
```

SPRACHSPEZIFISCHES

- Tiefensuche
- [ ] - Array aus rekursiver Prozedur

```
/recfind {
    dup visited exch 1 put
    % alle nachbarn des aktuellen knoten
    dup getgraph exch get {
        % wurde der knoten schon besucht?
        dup visited exch get 0 eq {
            % ist der knoten das ziel?
            dup destinationnode eq {
                ] dup [ exch
                aload pop
            } {
                recfind
            } ifelse
        } if
        pop
    } forall
    dup visited exch 0 put
} def
```

SPRACHSPEZIFISCHES

- [ ] - Array aus rekursiver Prozedur

```
% startknoten-- [[path]]
/paths {
  [ exch
  [ exch
    recfind
  cleartomark
  ]
} def
```

```
% proc [path0 path1 .. pathN] -- path
% proc: path0 path1 -- minPath
/findBestPath {
    dup 0 get exch
    {
        2 index exec
    } forall
    exch pop
} def
```

```
% path0 path1 -- pathMin
/dummySearch {
    pop
} def
```

```
% path0 path1 -- pathMin
/shortestSearch {
    2 copy length
    exch length
    lt {
        exch
    } if
    pop
} def
```