

Von Labyrinthen zu Algorithmen

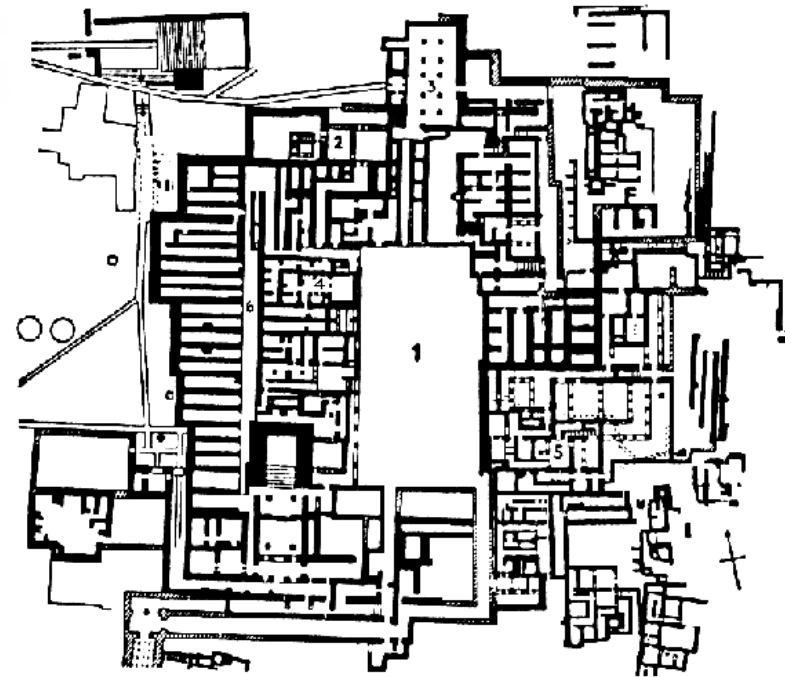
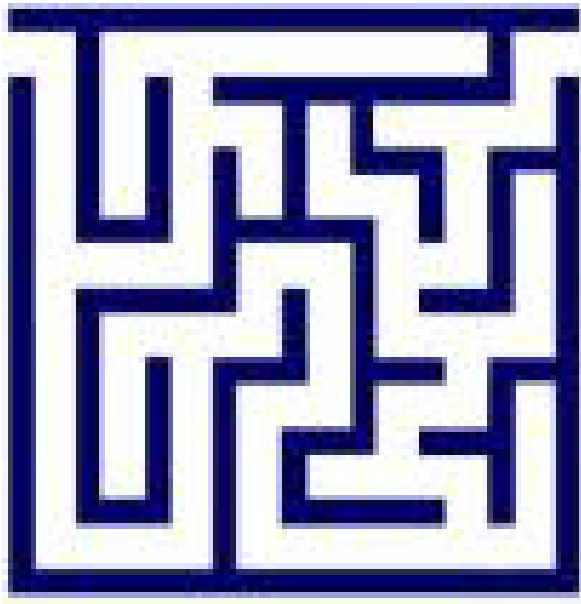
Gerald Futschek

Wie kommt man aus einem Labyrinth heraus?

- **Labyrinth** (griechisch: Haus der Doppelaxt, wahrscheinlich Knossos auf Kreta)



Labrys

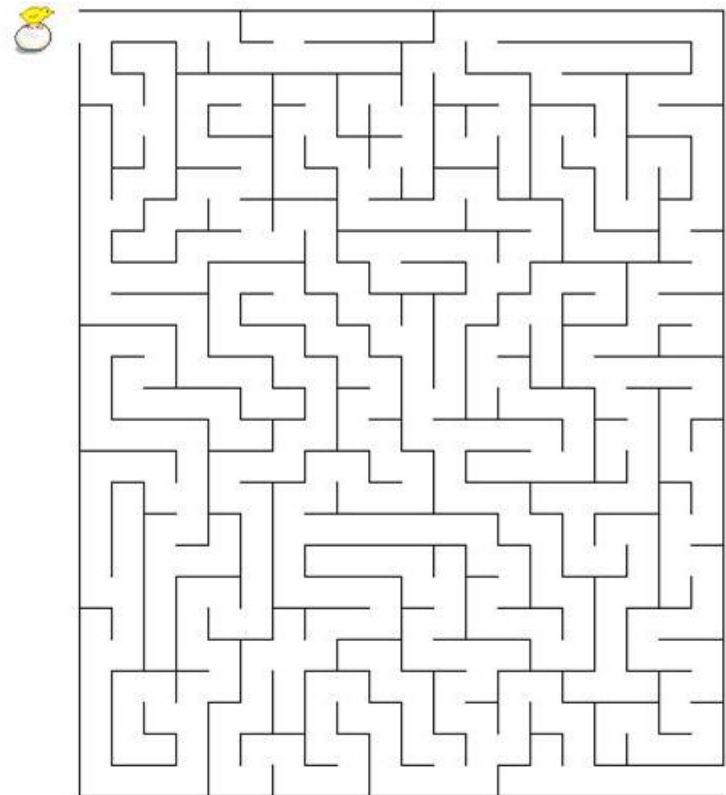


Grundriss des Palastes von Knossos

Fragestellungen zu Labyrinthen

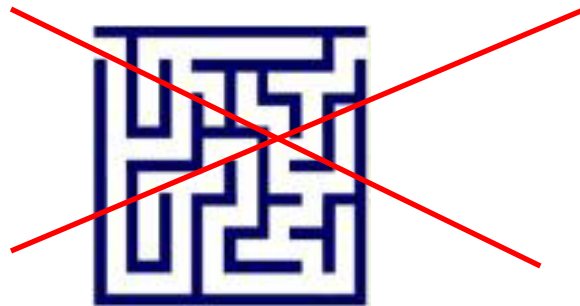
- Finde einen Weg durch das Labyrinth
- Finde einen Weg hinaus
- Finde einen Weg zu einem bestimmten Punkt im Labyrinth
- Gibt es vielleicht mehrere Wege?
- Welcher Weg ist der kürzeste?
- Wie findet man solche Wege?

Aide le poussin à retrouver sa maman.

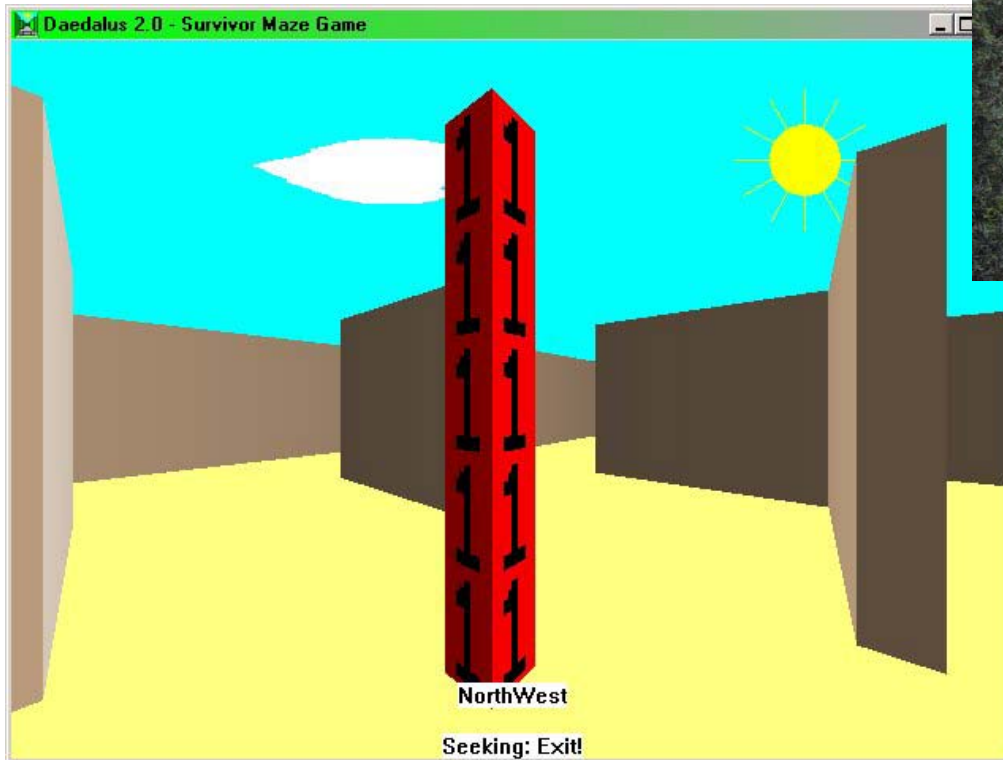


Präzisierung der Aufgabenstellung

- Gesucht ist ein Weg von Position A nach B
- Form, Größe und Struktur des Labyrinths ist zunächst nicht bekannt



Sichtweise in einem Labyrinth







Suche nach einem Weg

- An jeder Kreuzung: Welchen Gang soll man gehen?
 - einen zufälligen?
 - einen bestimmten Gang?
 - alle Gänge der Reihe nach?
(wie geht das genau?)

Strategie gesucht!

Eine Idee für eine Strategie

- „Man geht immer der linken Wand entlang!“
- Diese Strategie funktioniert jedenfalls sicher von einem Eingang zu einem Ausgang (Spiegelkabinett)



Abstraktion

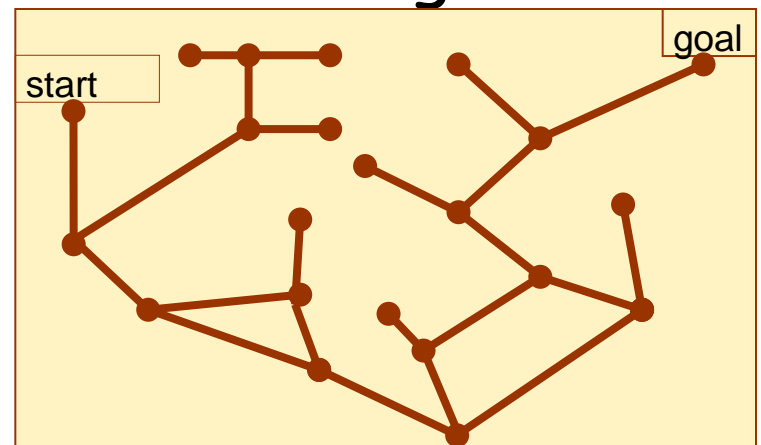
- Labyrinth besteht aus
 - Kreuzungen und
 - Gängen zwischen Kreuzungen
- Länge und Form der Gänge nicht wichtig

Modellierung als **Graph**:

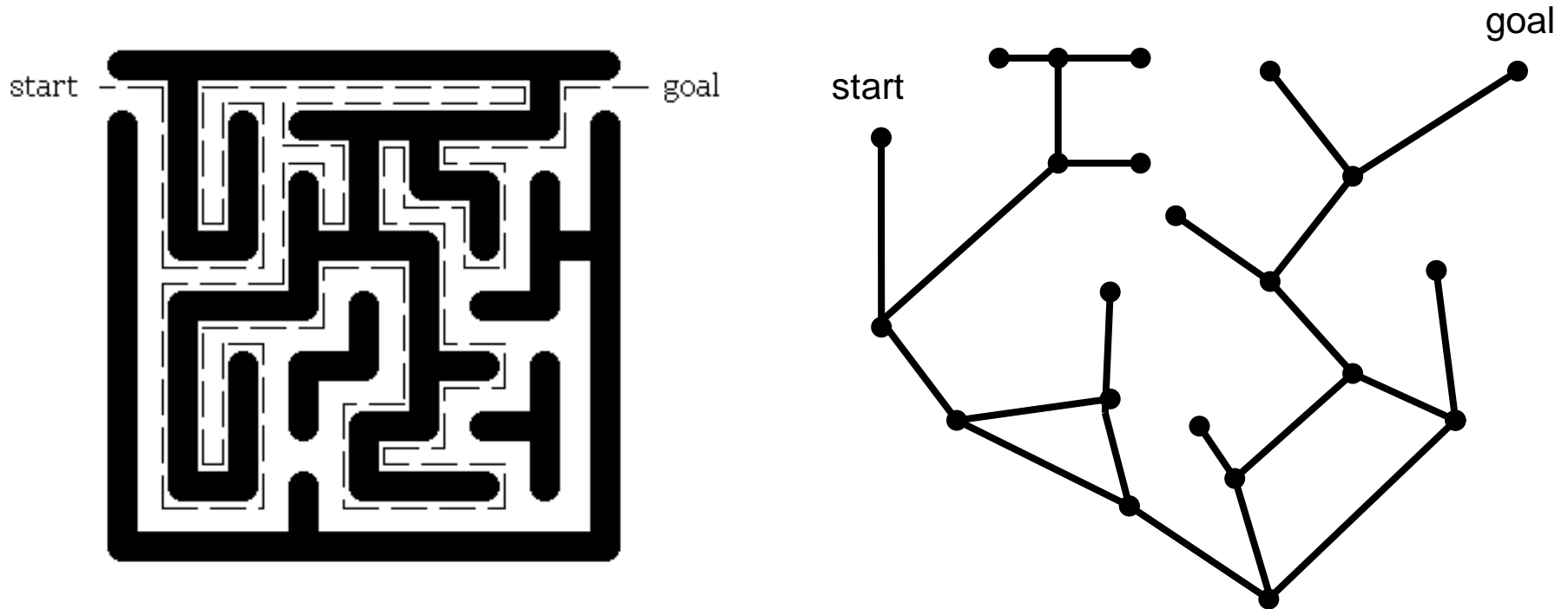
Knoten und Kanten

Kreuzungen werden zu Knoten

Gänge werden zu Kanten



Labyrinth - Graph



Was entspricht der linken Wand in einem Graphen?

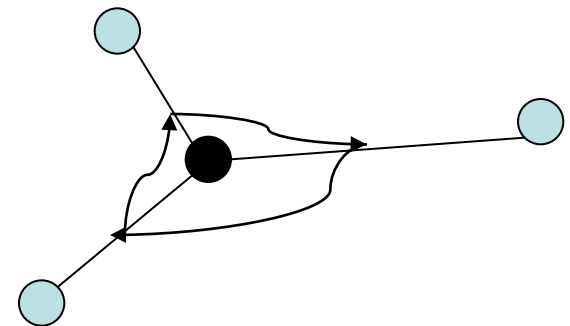
Modellierung der Reihenfolge der Gänge

- Ein Graph beschreibt nur Knoten und welche Knoten durch Kanten verbunden sind.
- In einem Graph gibt es keine Reihenfolge der Kanten

Für jeden Knoten müssen wir eine Reihenfolge der Kanten zusätzlich modellieren.

Da alle Kanten „gleichberechtigt“ sind, definieren wir gleich eine **zyklische Ordnung**:

Zu jeder Kante e eines Knoten gibt es eine Nachfolgerkante $\text{succ}(e)$



$\text{succ}(e)$ ist der
1. Gang von links!

Modellierung als Embedded Graph

- Ein **Embedded Graph** (in die Ebene eingebettet) hat die Kanten jedes Knoten zyklisch geordnet
- dh. man kann, wenn man über eine Kante e zu einem Knoten kommt, mit $\text{succ}(e)$ den 1. Gang von links bestimmen
- Mit der zyklischen Ordnung hat man auch eine Reihenfolge, um alle Gänge einer Kreuzung systematisch zu durchlaufen!

Grundoperationen für Labyrinth

Die folgenden Grundoperationen dürfen in einem Labyrinth-Algorithmus verwendet werden:

Man kommt stets von einem Gang an eine Kreuzung:

- Abfrage: Anzahl weiterer Gänge bei dieser Kreuzung? (Sackgasse bei null weiteren Gängen)
- Aktion: Man wählt den i -ten Gang von links und geht in diesem Gang bis zur nächsten Kreuzung
- Aktion: Man geht den Gang, den man soeben gekommen ist, bis zur letzten Kreuzung zurück
- Abfrage: Ziel erreicht? Ja/Nein

Kann man mit diesen Grundoperationen einen Weg von A nach B im Labyrinth finden? Wie?

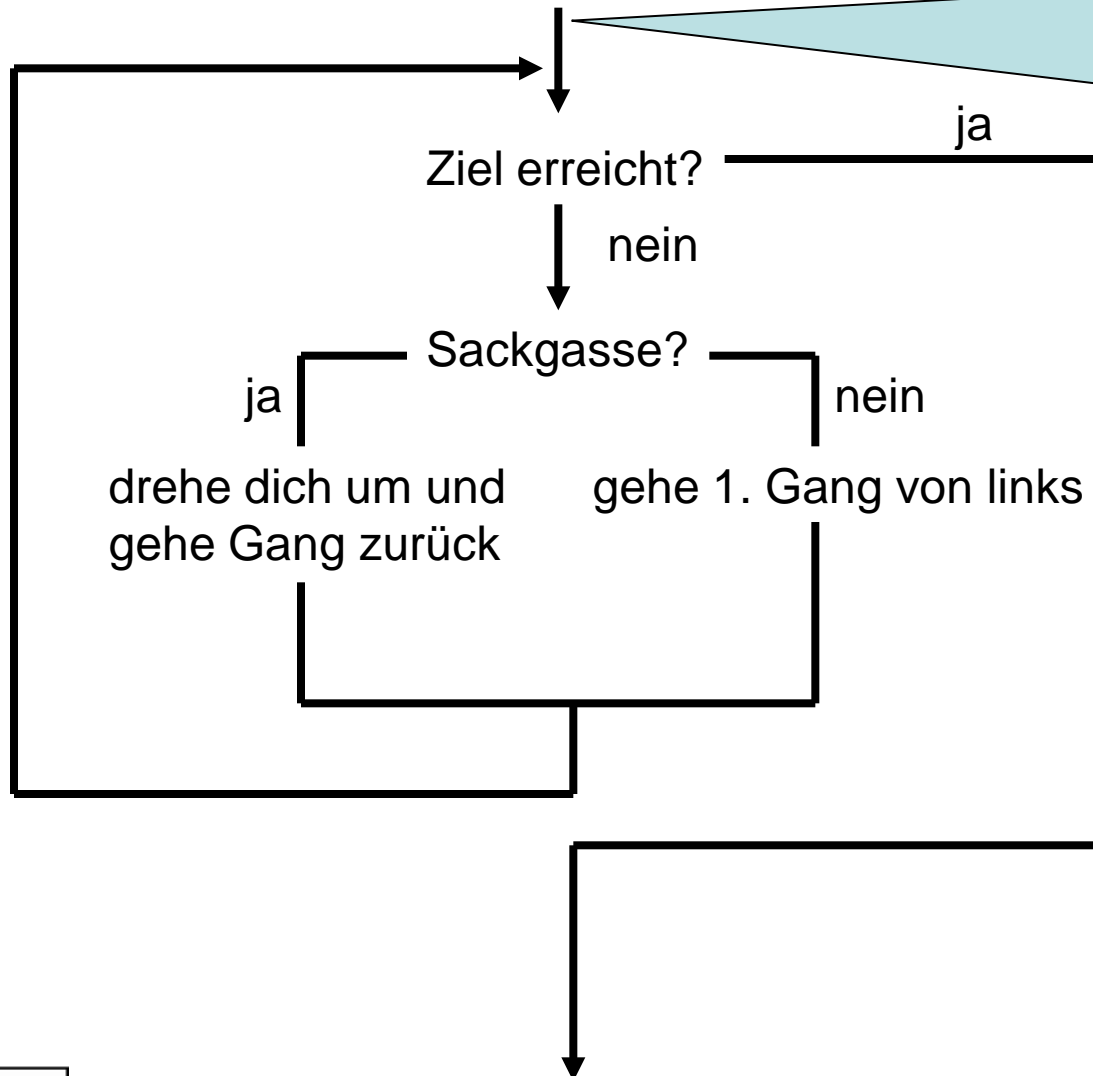
Linke Wand entlang

- Wie formuliert man den Algorithmus mit den Grundoperationen?
- Es gibt ja in der Abstraktion (Graph) keine Wände, sondern nur mehr Kanten!

Algorithmus mit den 4 Grundoperationen:

solange Ziel nicht erreicht
falls Sackgasse
 drehe dich um und gehe Gang zurück
sonst
 gehe 1. Gang von links

Flussdiagramm



Welche Anfangsbedingungen müssen erfüllt sein, damit dieser Algorithmus terminiert?

Precondition eines Algorithmus

- **Precondition** (Vorbedingung, Anfangsbedingung)
muss vor dem Algorithmus erfüllt sein, damit er terminiert und die gewünschten Ergebnisse liefert

Precondition

Algorithmus

Preconditions für Linke Wand Algorithmus:

- Ziel muss erreichbar sein
- Am Weg zum Ziel ist kein Zyklus

solange Ziel nicht erreicht
falls Sackgasse
 drehe dich um
 und gehe Gang zurück
sonst
 gehe 1. Gang von links

Postcondition eines Algorithmus

- **Postcondition**

(Nachbedingung,
Endbedingung)

beschreibt den gewünschten
Endzustand des Algorithmus

Algorithmus

Postcondition

**Postcondition des
Linke Wand
Algorithmus:**

Ziel ist erreicht

solange Ziel nicht erreicht
falls Sackgasse
 drehe dich um
 und gehe Gang zurück
sonst
 gehe 1. Gang von links

Bedeutung der Precondition

- **Precondition ist erfüllt:**

Algorithmus terminiert sicher und nachher ist die Postcondition erfüllt

Precondition

Algorithmus

- **Precondition ist nicht erfüllt:**

Es ist nicht garantiert, dass der Algorithmus terminiert oder nachher die Postcondition erfüllt ist.

Postcondition

In diesem Fall soll der Algorithmus gar nicht ausgeführt werden!

Problem der Zyklen

- Die Strategie „Linke Wand entlang“ funktioniert leider nicht bei allen Labyrinthen, wenn man von A nach B will!
- Möglicherweise gibt es Zyklen, sodass es auch unendlich lange Pfade gibt (das Verfahren terminiert nicht!)
- **Gibt es trotzdem Strategien, zum Ziel zu kommen?**

