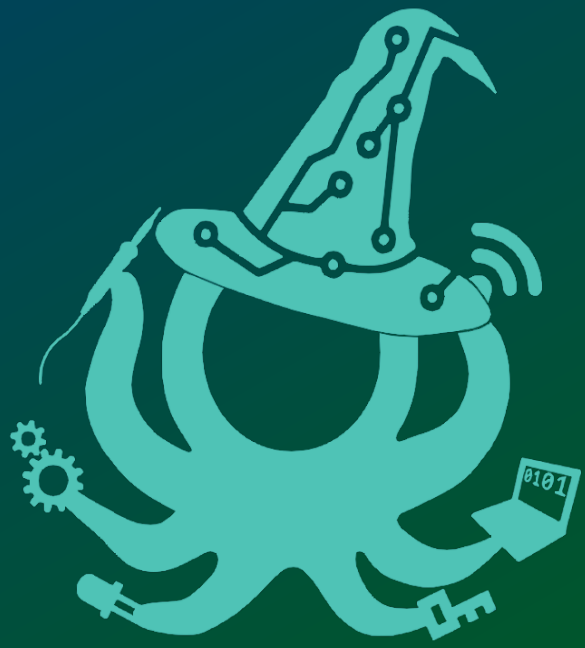
Haecksathon #9



Wie Computer Spiele spielen

fisk

2026-08-18

Was sind Spiele?

Beispiele

BeiSpiele

- Go
- Schach
- Tic-Tac-Toe
- Vier-gewinnt

GegenbeiSpiele

- Poker, Black Jack, ...
  (nicht deterministisch)
- LoL, DotA, CS, SC, ...
  (nicht rundenbasiert)
- Memory
  (keine perfekte Information)

Grundannahmen

Zur Vereinfachung ein paar Grundannahmen:

- 2 Spielende
- kompetitiv (also gegeneinander)
- rundenbasiert und abwechselnd
- deterministisch (also kein Zufall)
- perfekte Information

Definitionen

- **Spielzustand**: Aktueller Zustand des Spiels mit allen nötigen Informationen
  - Das Tic Tac Toe Feld, mit der zusätzlichen Information wer gerade am Zug ist.
- **Zug**: formale Interaktion der Spielenden mit dem Spiel
  - Ein Kreuz oder Kreis in ein freies Feld malen.
- **Endzustand**: Ein Spielzustand, in welchem das Spiel entschieden ist, also gewonnen oder verloren wurde oder unentschieden ist oder aus anderen Regelgründen als beendet erklärt wurde.
  - Es gibt drei Symbole in Folge -> entsprechende Spielende hat gewonnen.
  - Volles Spielfeld -> unentschieden.

Einfachere Ansätze

Einfachere Ansätze

- **Probabilistisch**: zufälligen Zug spielen
- **Greedy**: spiele den Zug, welcher sofort den meisten Vorteil bringt
- **Brute force**: rechne den optimalen Zug aus
- **Regelbasiert / heuristisch**: Antworten für Standartsituationen, wenn x passiert dann y,
  z.B. **wenn** es zwei Symbole in einer Linie gibt, **dann** setze das dritte
- **Look-up-table**: große Tabelle, wo für jeden Spielzustand steht, welcher Zug gemacht wird
- Kombinationen aus diesen Ansätzen

Suchalgorithmen

grobe Erklärung

Der Spielbaum ist ein gerichteter Graph, welcher den Ablauf des Spiels darstellt.

- Knoten: Spielzustände

Spielzustand A

- Wurzel: Startzustand des Spiels
- Kanten:

Vom Spielzustand A führt der Zug k zum Spielzustand B.

Beispiel

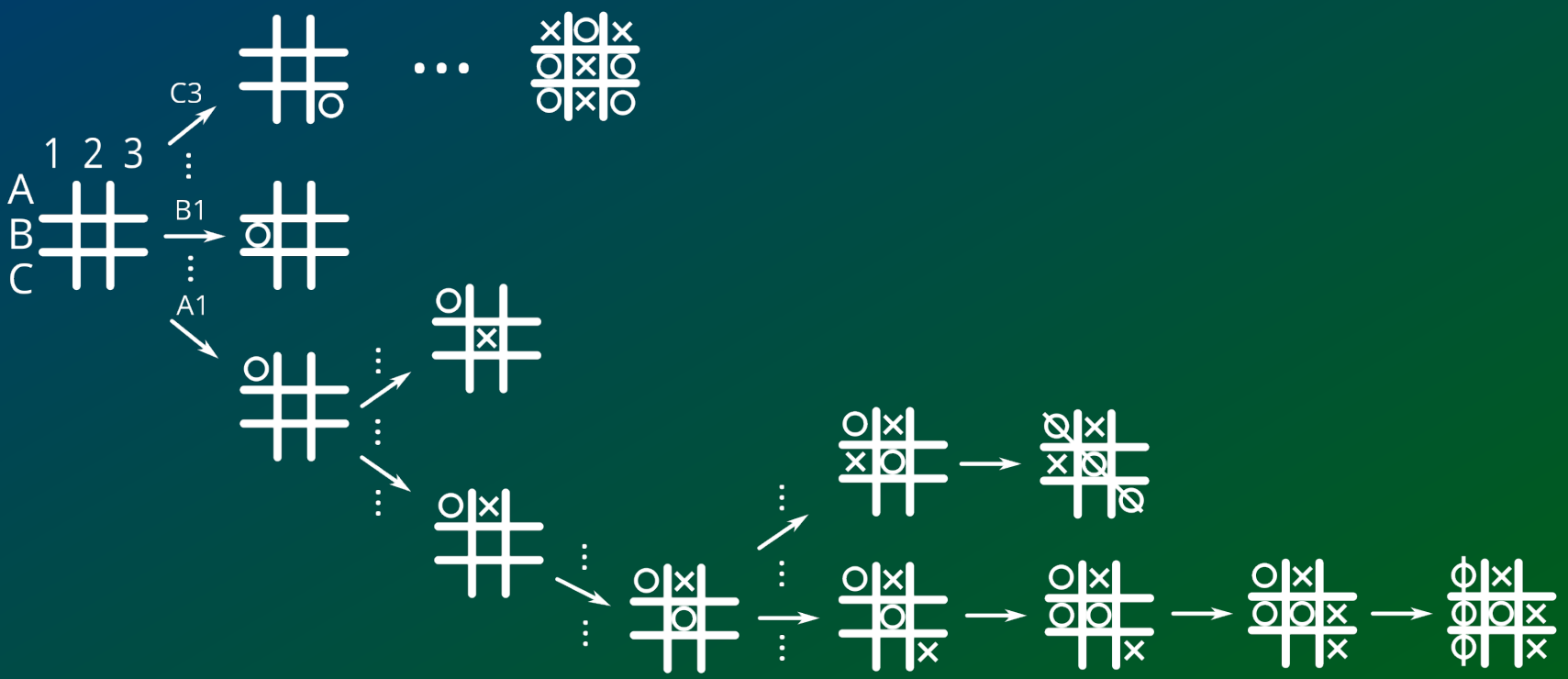


Figure 1: Bild eines Tic-Tac-Toe Spielbaums

Verzweigungsgrad (Branching Faktor)

- Ausgangsgrad der Spielzustände, also wie viele erlaube Züge es durchschnittlich gibt.

Spiel	Verzweigungsgrad	log_{10}(Zustände)
Dame (8x8)	2.8	18
Go (19x19)	250	170
Schach	35	44
Shogi	92	71
Tic-Tac-Toe	4	3

Minimax

Basics

grobe Erklärung

- Tiefensuche (mit maximaler Tiefe k) im Spielbaum
  - Spielbaum wird bis zu dieser Tiefe komplett exploriert.
- Falls Endzustand oder Tiefe k erreicht wird, werden Zustände bewertet und die Bewertung wird wieder *nach oben gegeben*.

Bewertungsfunktion

Die Bewertungsfunktion ist eine Funktion, welche einem Spielzustand bewertet.

Also irgendetwas, dass sagt wie gut ein Zustand jeweils für die Spielenden ist. Z.B.:

- bereits geschlagene Figuren
- Position auf dem Spielfeld
- schon sichere Punkte

Anmerkung zu Implementierung

Dafür muss aber eine ganz spezielle Bewertungsfunktion existieren:

- Funktion bewertet den Spielzustand unabhängig davon, wer am Zug ist.
- Jeweils eine der Spielenden möchte diesen Score maximieren (und die andere möchte minimieren).

Beispiel

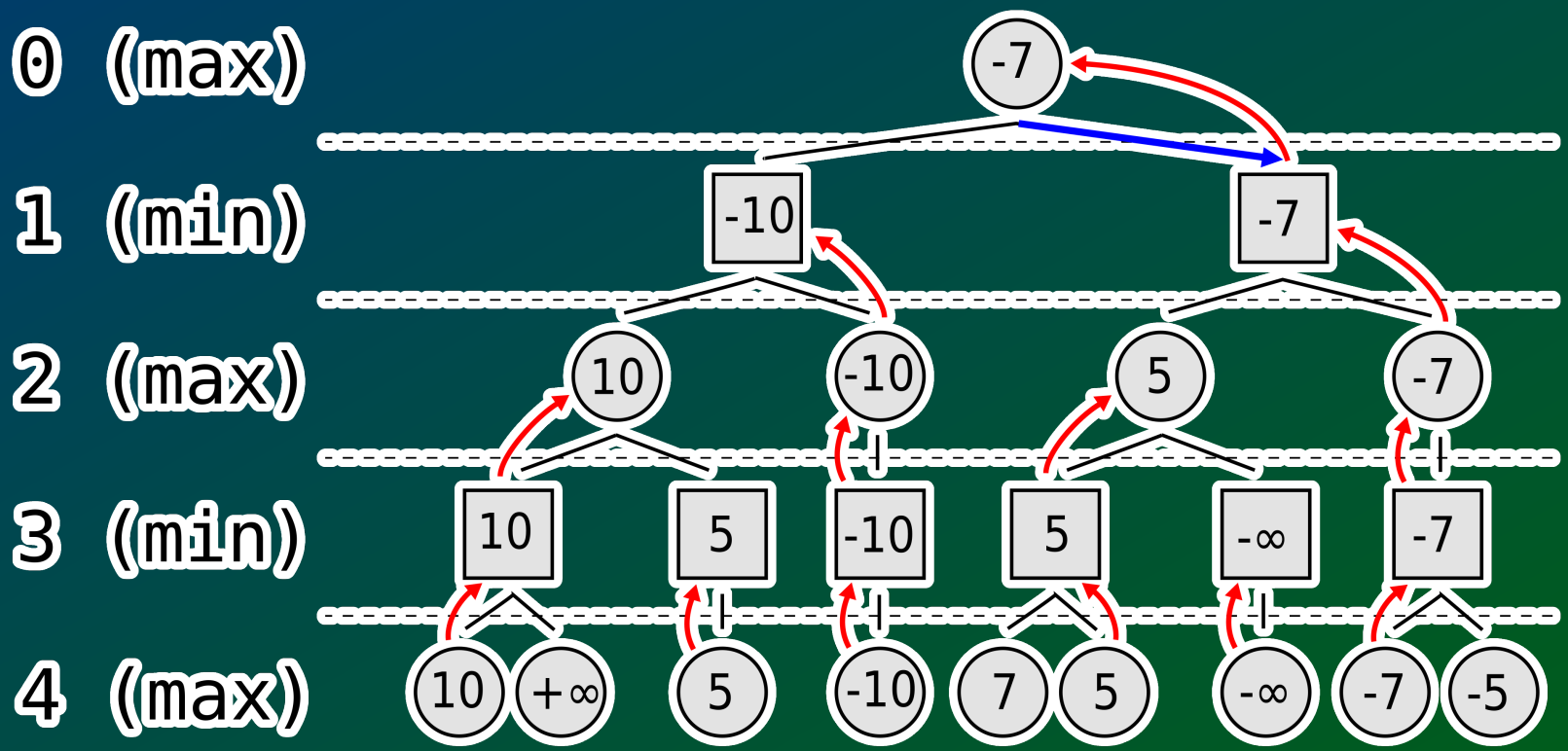


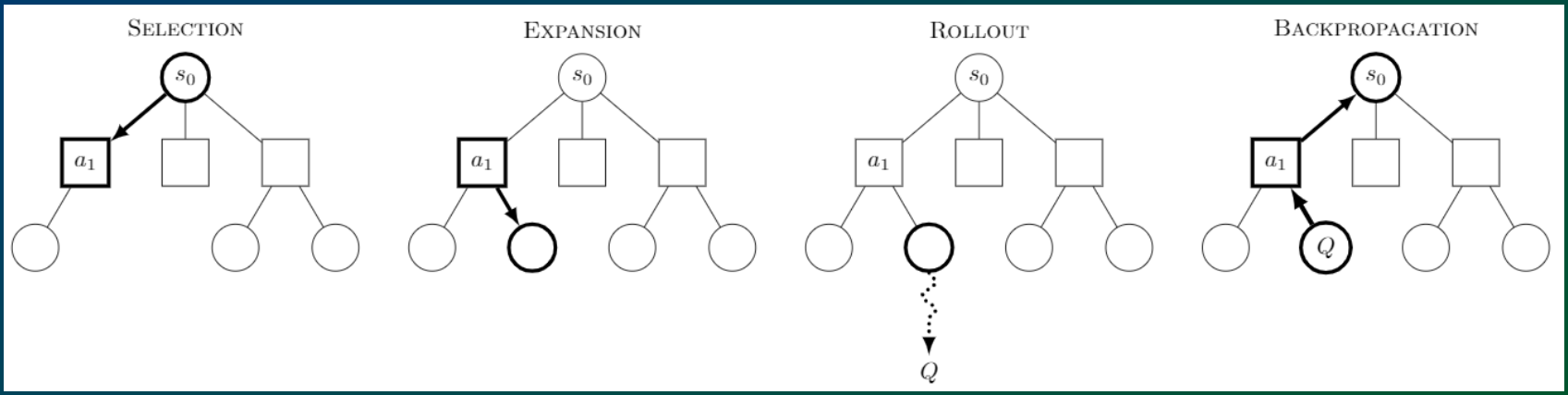
Figure 2: Minimax computation example

Monte-Carlo-Tree-Search MCTS

Basics

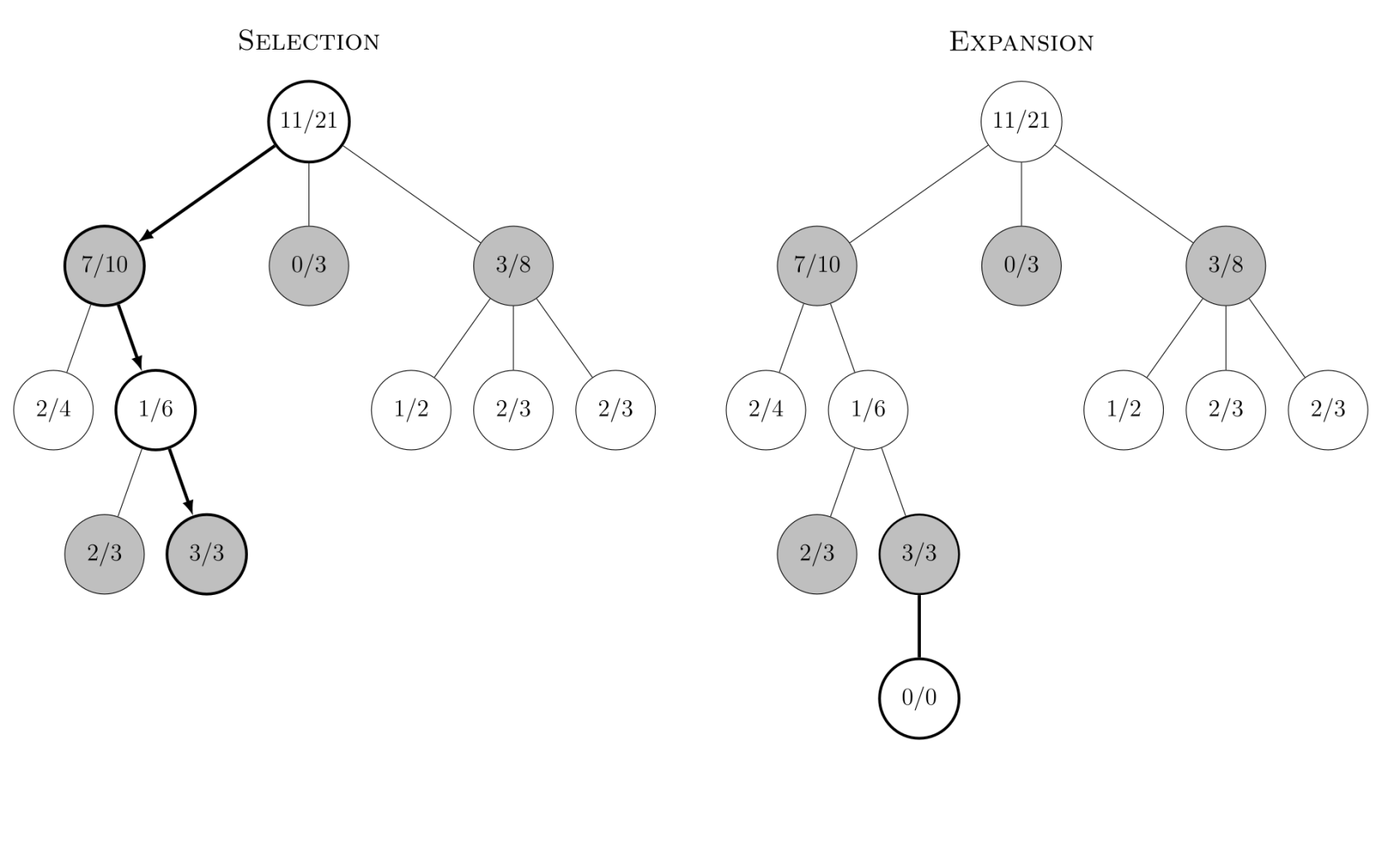
grobe Erklärung

Spiele zufällig Züge bis Spiel vorbei ist. Merke dabei wie oft gewonnen wurde und bewerte Spielzustände dementsprechend.

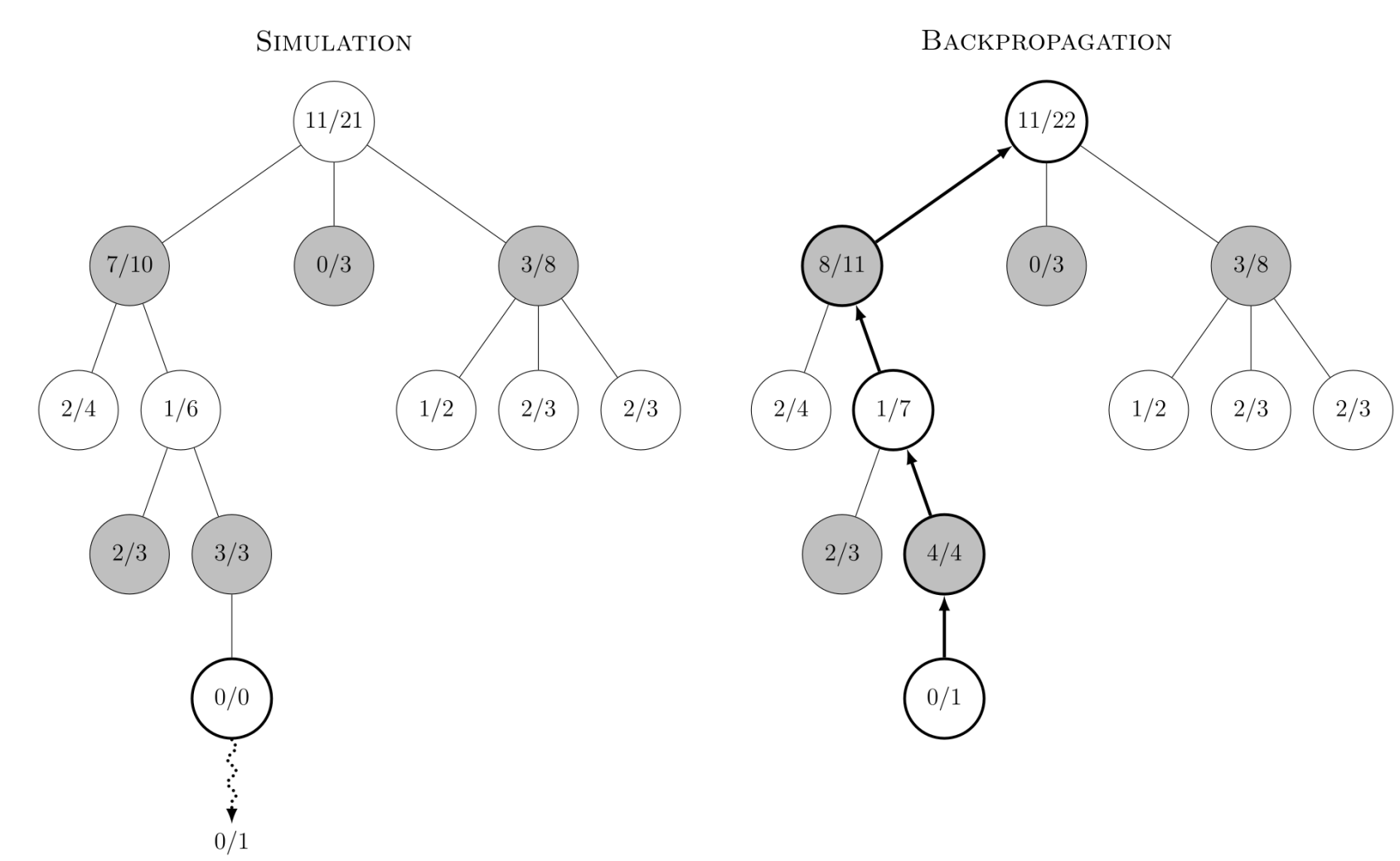


1. Zustand merken (im gemerkten Spielbaum)
2. Nachfolger generieren (im gemerkten Spielbaum)
3. Ausrollen
   - (wiederholt) zufälligen Zug spielen, bis Spiel entschlossen ist
4. Anpassung der Wahrscheinlichkeiten (im gemerkten Spielbaum)

Selection + Expansion



Simulation + Backpropagation



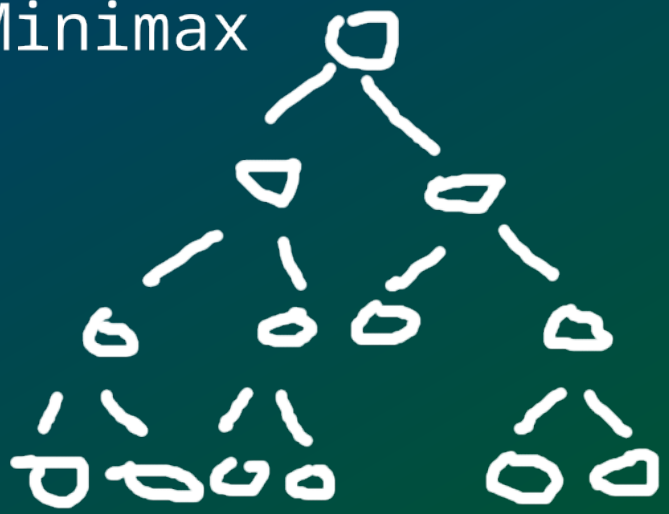
Zusammenfassung

Tipps für den Anfang

- Erstmal einfachere Ansätze ausprobieren lohnt sich!
  - sind mindestens ein guter Vergleich
- Mit vielen kleinen Iterationen und Anpassungen planen
- auf Formalisierung achten, Details sind hier oft extrem wichtig
- **TESTEN!**

Suchbaumstruktur

Minimax



MCTS

Eigenschaften

Minimax

- Braucht Zustandsbewertung
- Laufzeit sehr stark abhängig vom Verzweigungsgrad
- Exploriert kompletten Spielbaum (bis zu Tiefe k)
- Spielt (auf kurze Sicht) optimal
- gut kombinierbar mit Heuristiken

MCTS

- braucht keine Zustandsbewertung
- lässt sich dynamisch beenden und Ablauf ist gut anpassbar
- merkt wenig Spielzustände, diese oft sehr asymmetrisch
- sehr gut kombinierbar mit Schätzung der Zustandsbewertung

Abwägungen

Minimax

- genaue Bewertungsfunktion
- Endspiel
- Spiel hat weniger Langzeit Strategien
- Verzweigungsgrad sehr niedrig
- gute Möglichkeiten die Suche einzuschränken

MCTS

- Spiel terminiert *wahrscheinlich*
- keine Bewertungsfunktion oder sie ist nicht einfach zu berechnen, ungenau, ...
- Verzweigungsgrad sehr hoch
- dynamische Anpassbarkeit des Ablaufs ist relevant

generell gilt aber: **Testen!**

Danke!

Danke!

Danke an die vielen Unterstützenden und Helfenden!

Fragen

References

References

- Complexities of some well-known games (Wikipedia Artikell, abgerufen 2026-02-12, referenziert die entsprechenden Paper)
- MCTS Steps images
  - https://en.wikipedia.org/wiki/File:MCTS_Algorithm.png
  - https://en.wikipedia.org/wiki/File:MCTS-steps.svg
- Minimax Example
  - https://en.wikipedia.org/wiki/File:Minimax.svg
- Haecksen Logo & Colors
  - https://wiki.dezentrale.space/Dezentrale/Design