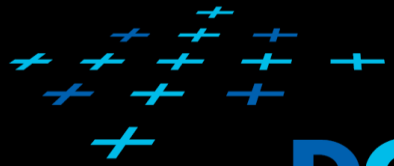


DCGI

KATEDRA POČÍTAČOVÉ GRAFIKY A INTERAKCE

Umělá inteligence pro hry

Jiří Bittner

Obsah přednášky

- Úvod — AIG 1.1 – 1.3
- Rozhodování (decision making) — AIG 5.1 – 5.4
- Plánování cest — AIG 4.1 – 4.3
- Řízení (steering) — AIG 3.2 – 3.4

AIG: I. Millington, J. Funge. Artificial Intelligence for Games 2nd ed., CRC Press 2009

Co je umělá inteligence (AI)?

- “Výpočetní systémy, které vykazují inteligenci”

- “Stroje napodobující kognitivní schopnosti lidí (řešení problémů, učení, …)”

- Game theory, computer vision, pattern recognition, deep learning, …

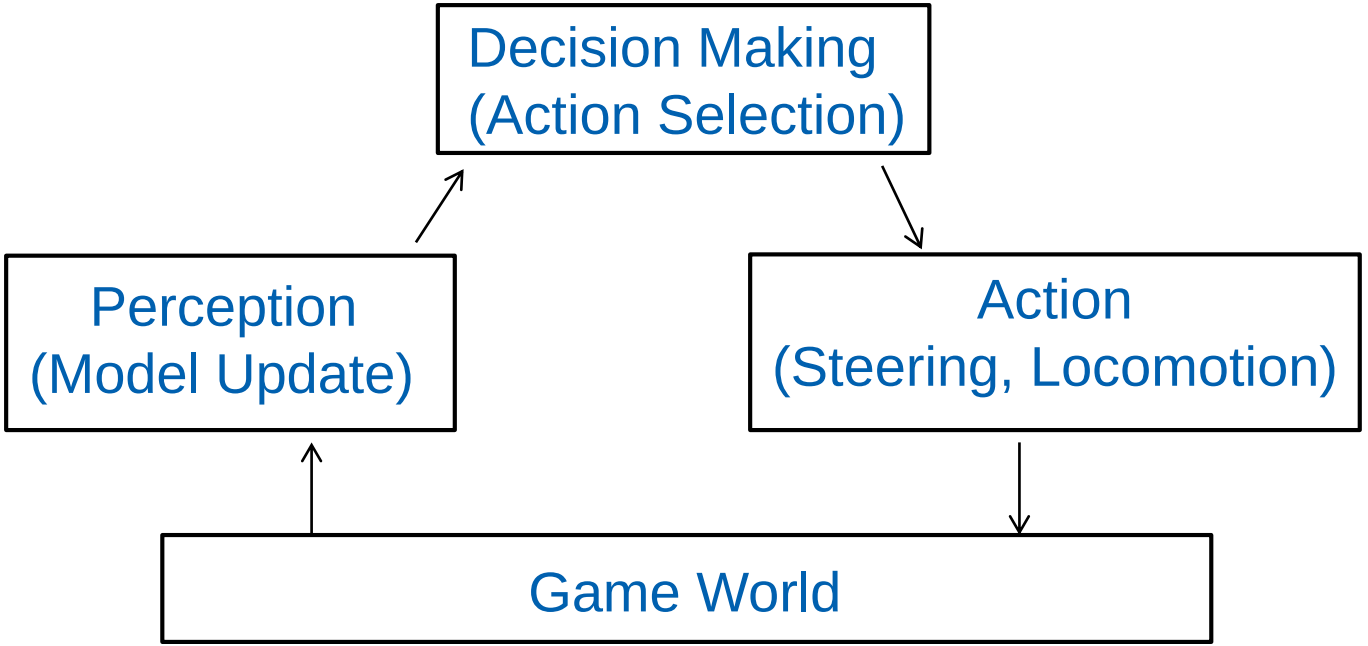
AI ve hrách

- Simulace chování NPC (non-player character)
  - Soupeři
  - Spoluhráči
  - Pomocné a autonomní postavy
- Chování NPC
  - Reaguje na aktuální stav hry
  - Chování NPC není přímou akcí hráče
  - Není výsledkem fyzikální simulace
- NPC “z pohledu hráče dělá to co by dělal člověk nebo jiná bytost”
  - AI si všimneme hlavně když selže
- Další role AI
  - Nápověda, instrukce, komentáře, ovládání kamery

Požadavky na AI systém

- Chytrý (ale ne moc)
  - NPC nepodvádí (většinou), př. NPC nemá vidět skrz zeď
- Konzistentní
  - Predikovatelné chování
- Efektivní
  - Velké množství NPC
- Přizpůsobivý
  - Skriptování, změny chování

Chování NPC - AI smyčka



NPC jako agent

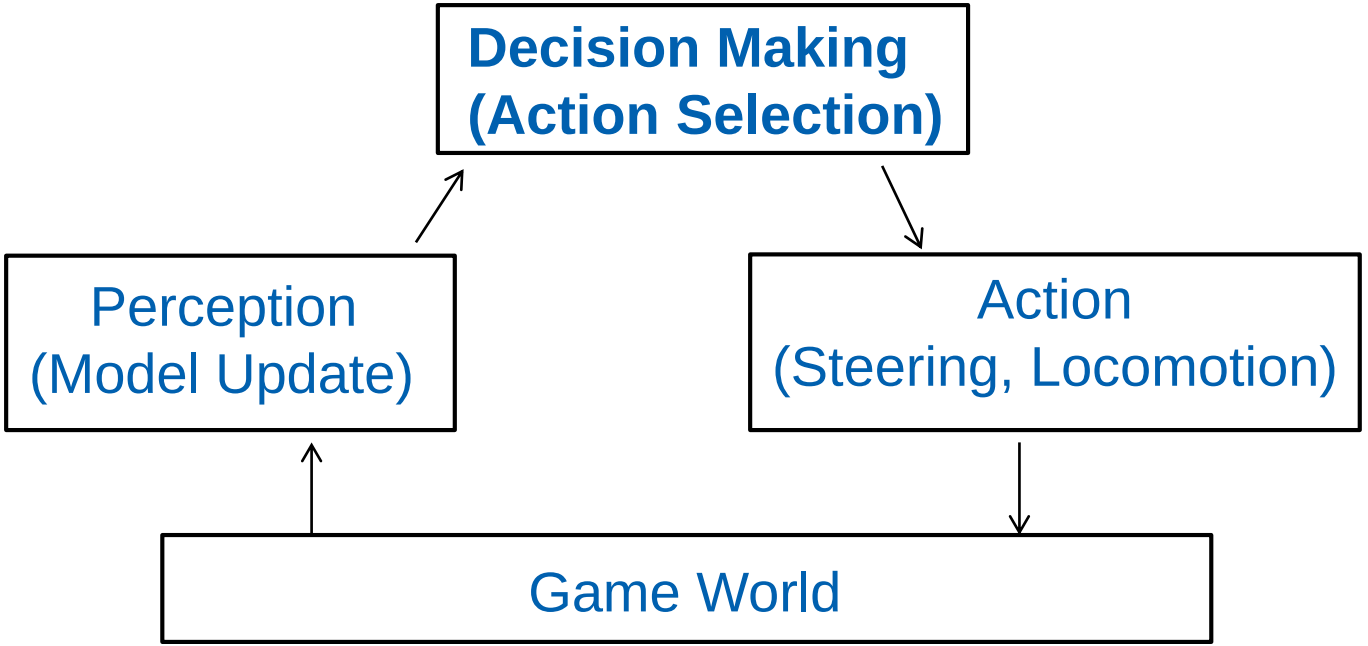
- Agent (pojem z oboru AI)
  - Reflexivní agent (if XX then YY)
  - Model-based agent (vniřní stav – FSM)
  - Goal-based (vzdálený cílový stav)
  - Utility-based (využitelnost různých cílových stavů)
  - Učící se agent

ZUI , 4. semestr

Obsah přednášky

Úvod	AIG 1.1 – 1.3
Rozhodování (decision making)	AIG 5.1 – 5.4
Plánování cest	AIG 4.1 – 4.3
Řízení (steering)	AIG 3.2 – 3.4

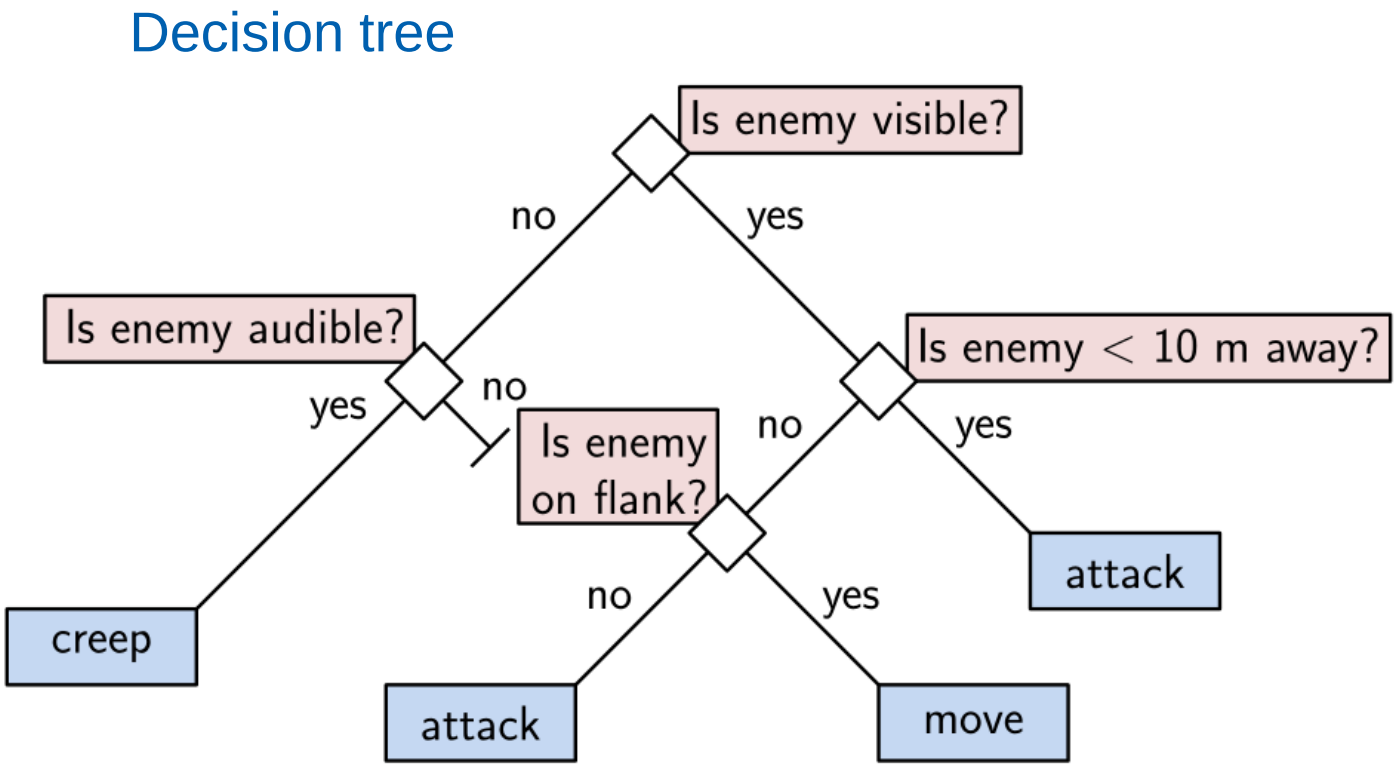
Rozhodování



Rozhodování – Decision Making

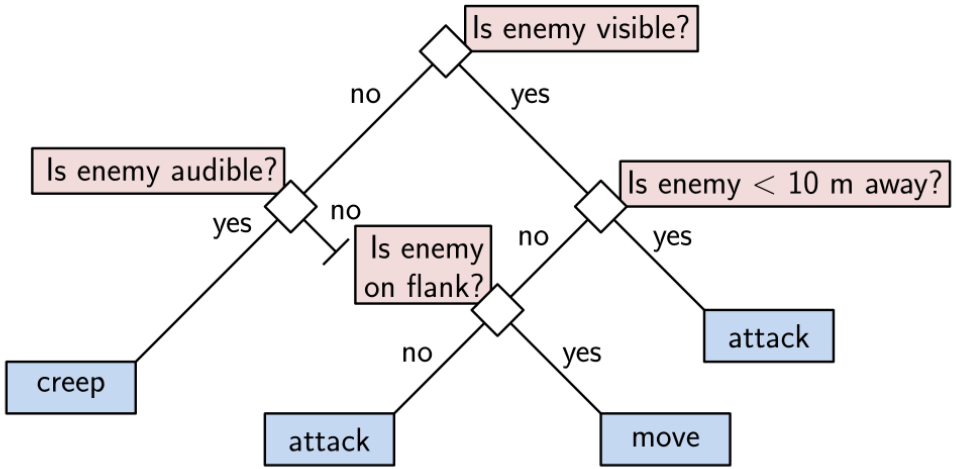
- Plánování vysokoúrovňových akcí NPC
- Rozhodování může být naprogramováno (LUA, C#)
  - Nepřehledné a těžko přenositelné know-how
  - Nemá vizuální podobu
- Metody
  - **Rozhodovací strom (decision tree)**
  - **Konečný automat (FSM)**
  - **Strom chování (behavior tree)**
  - Fuzzy sets
  - Fuzzy state machines
  - Rule based systems
  - Blackboards

Rozhodovací strom



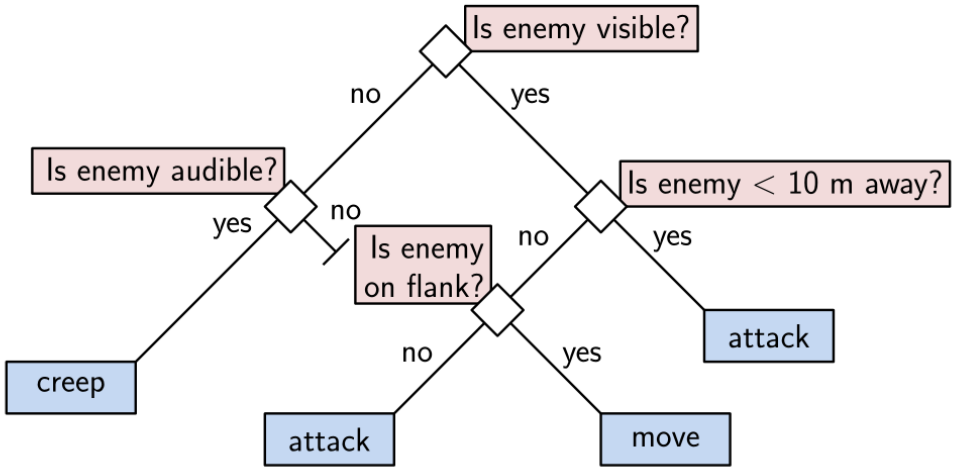
Rozhodovací strom - randomizace

- Náhodná rozhodnutí
  - Zvyšuje variabilitu chování
  - Definice pravděpodobností



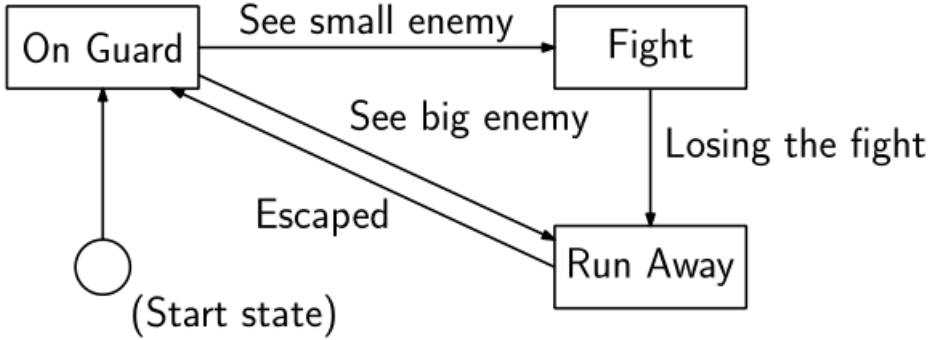
Rozhodovací strom

- Problém: nemá konzistentní vnitřní stav
- Použití na jednoduché modely chování

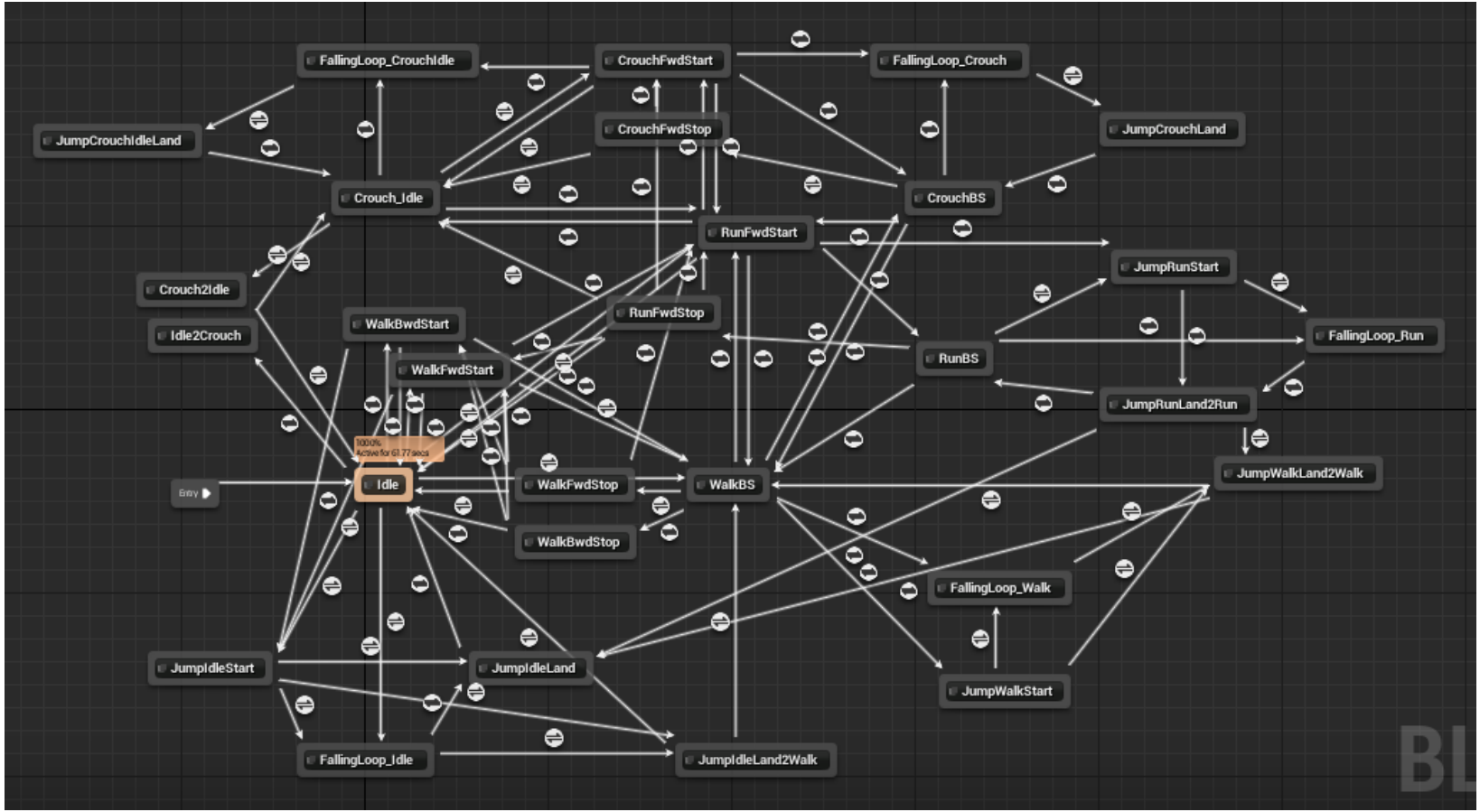


Konečný automat (FSM)

- Konečný počet stavů NPC (stav ~ vykonávání akcí)
- NPC má aktuální stav
- Přechod mezi stavy řízen vyhodnocením podmínek a událostí

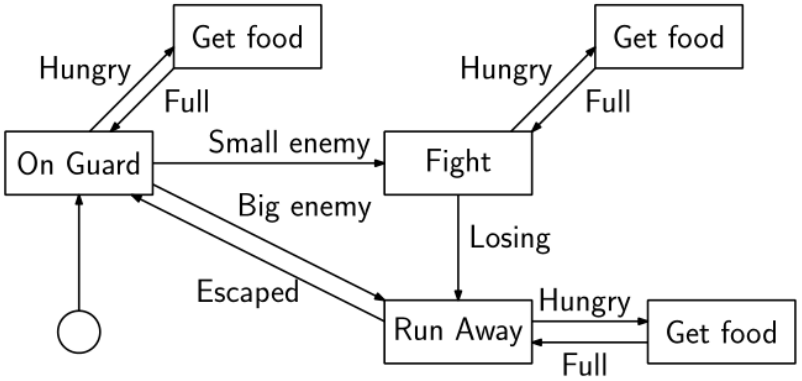


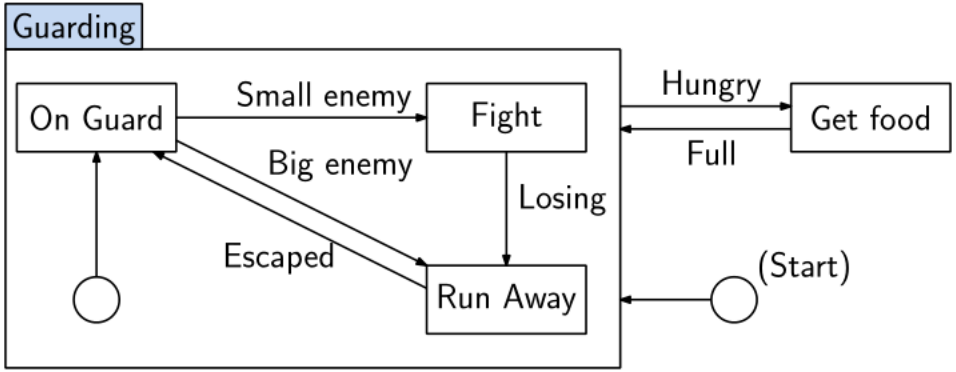
Konečný automat



Hierarchický FSM

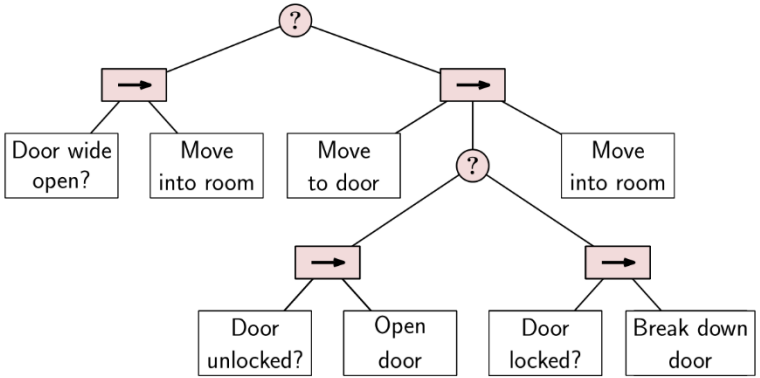
- Seskupení logicky souvisejících stavů do bloků
- Přechod mezi bloky stavů





Behavior Trees (Stromy chování)

- Jednotné rozhraní akcí, dotazů, skupin akcí (Halo 2, Bioshock, Spore)
- Hierarchické uspořádání
- Snadné využití komponent
- Vnitřní uzly mají vlastní logiku
  - Implementují způsob vyhodnocení podstromu (chování)

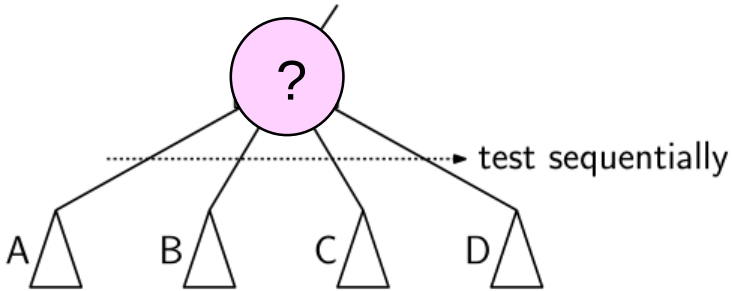


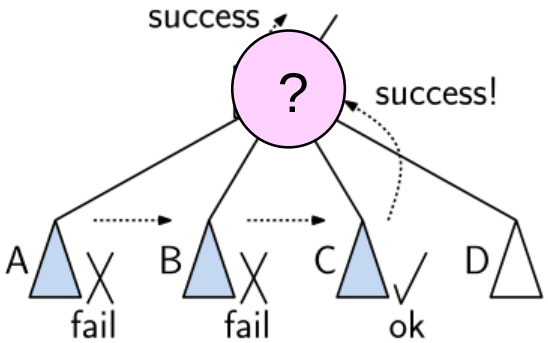
Behavior Trees

- Listy
  - **Podmínky** nebo **Akce**
- Vnitřní uzly
  - Způsob interpretace podstromu (**sekvence, selektor**, decorator)
- Strom procházen periodicky (např. každých 100ms)
  - Listy vrací jeden ze stavů: SUCCESS, FAIL, RUNNING
  - Stav je propagován nahoru ve stromu

Behavior Trees – Selector (Fallback)

- Najdi první úspěšnou akci (success / running)
  - Nutí vývojáře přemýšlet o úspěchu / neúspěchu !

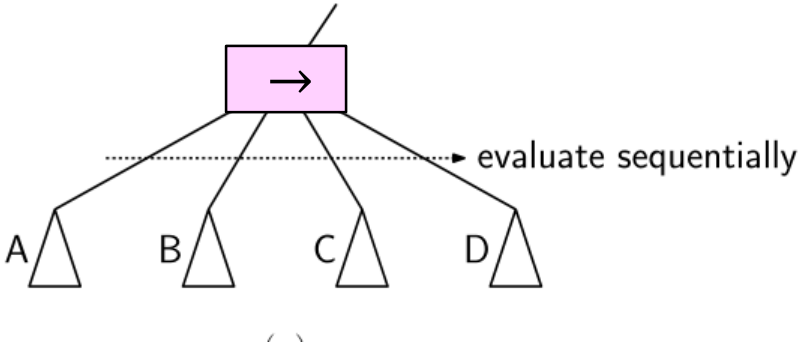


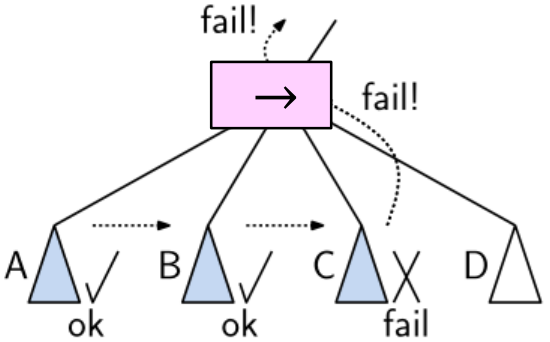



```
for i ← 1 to N do
  childStatus ← Tick(child(i))
  if childStatus == RUNNING then
    return RUNNING
  else if childStatus == SUCCESS then
    return SUCCESS
return FAIL
```

Behavior Trees - Sequence

- Vyhodnocuj dokud běží nebo neselže (running / fail)

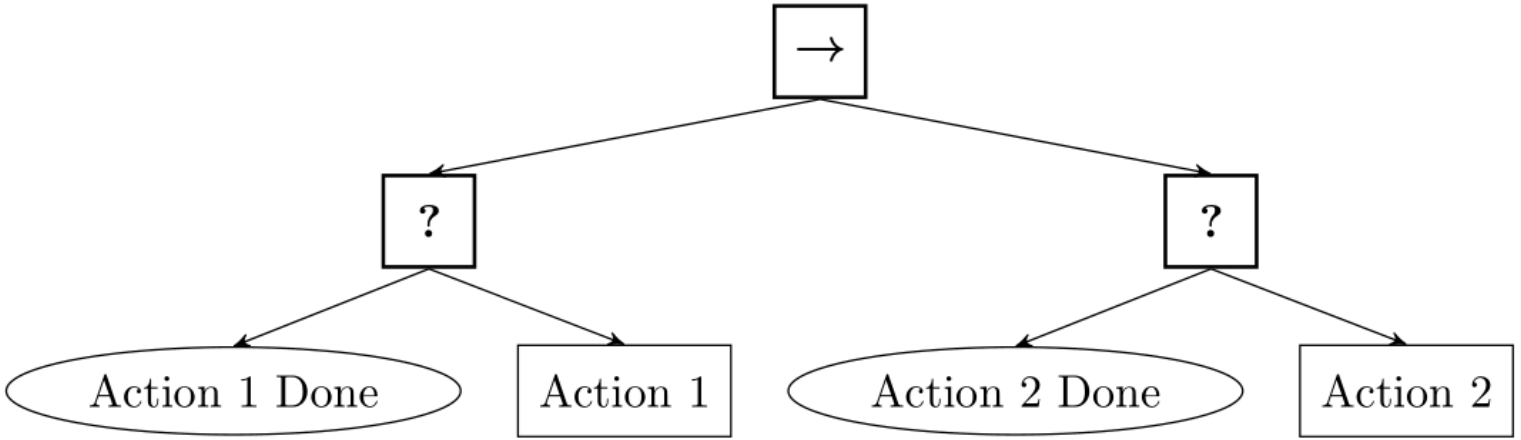





```
for i ← 1 to N do
  childStatus ← Tick(child(i))
  if childStatus == RUNNING then
    return RUNNING
  else if childStatus == FAIL then
    return FAIL
return SUCCESS
```

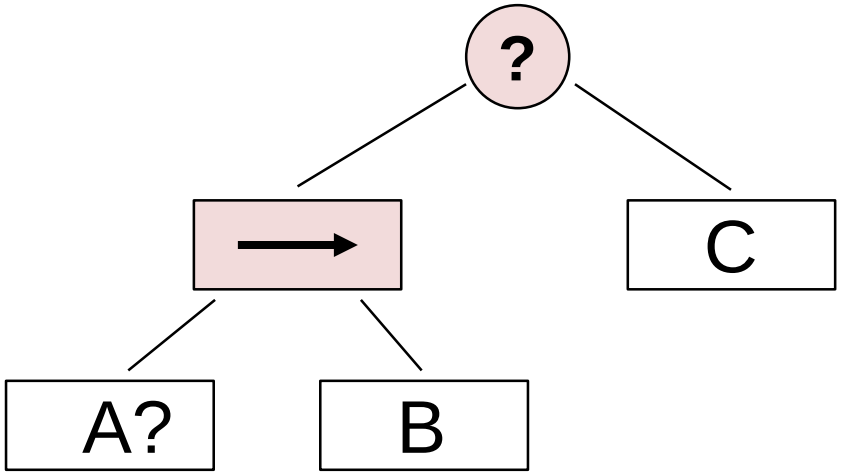
Behavior Trees - Listy

- **Podmínky** nebo **Akce**

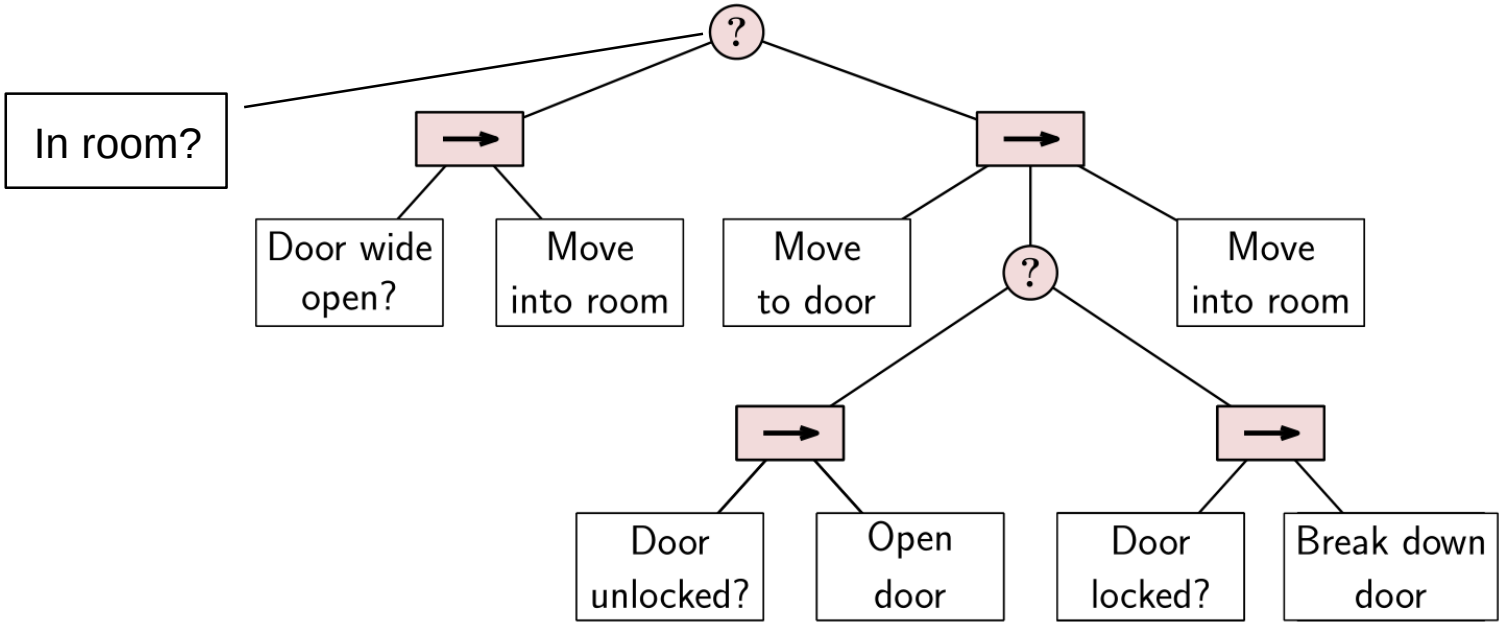


If-then-else?

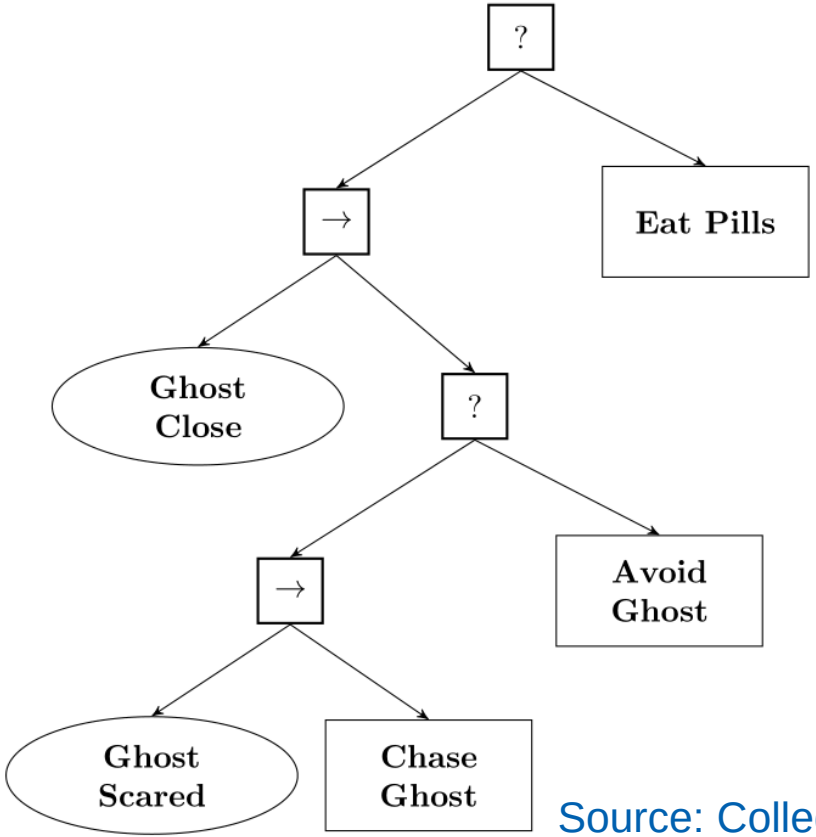
if A then B else C

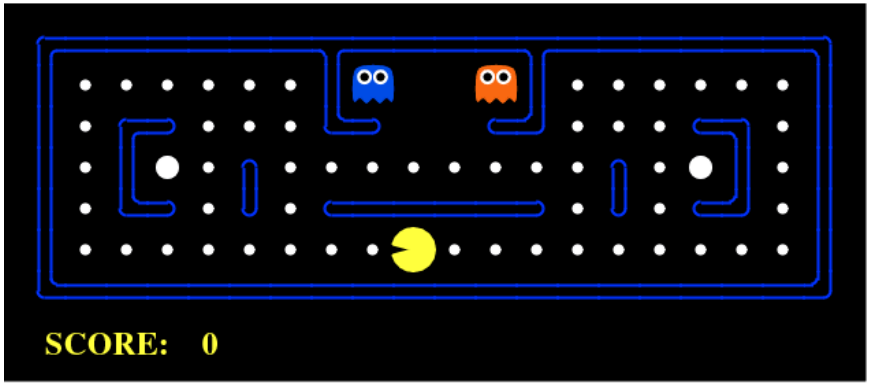


Příklad - Přechod do sousední místnosti



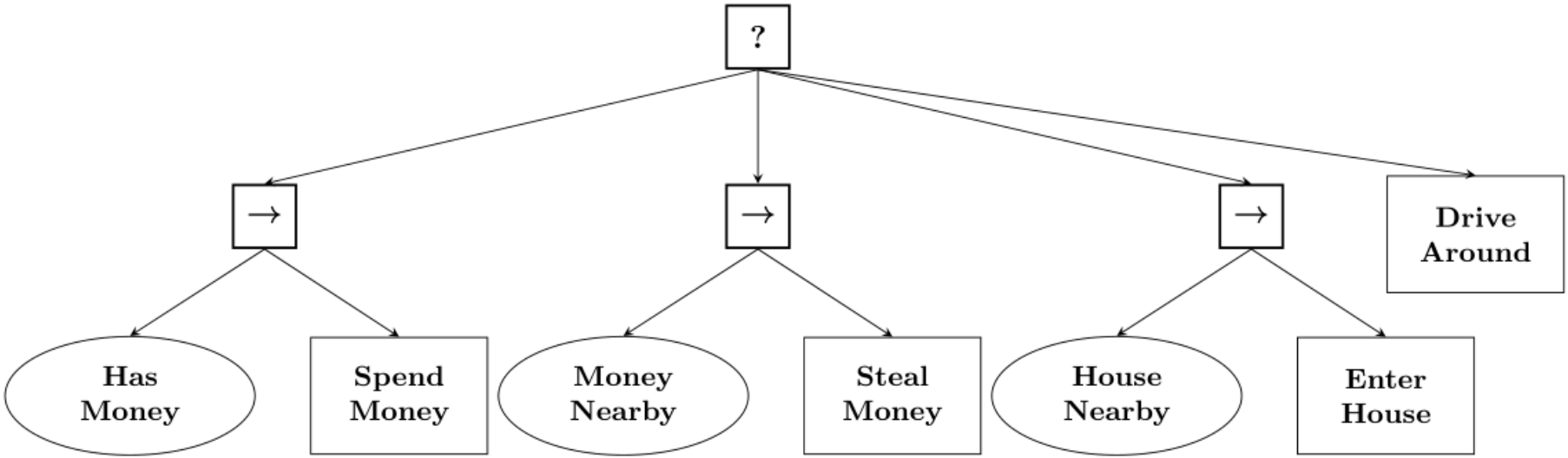
BT - Příklad (Pac-Man)



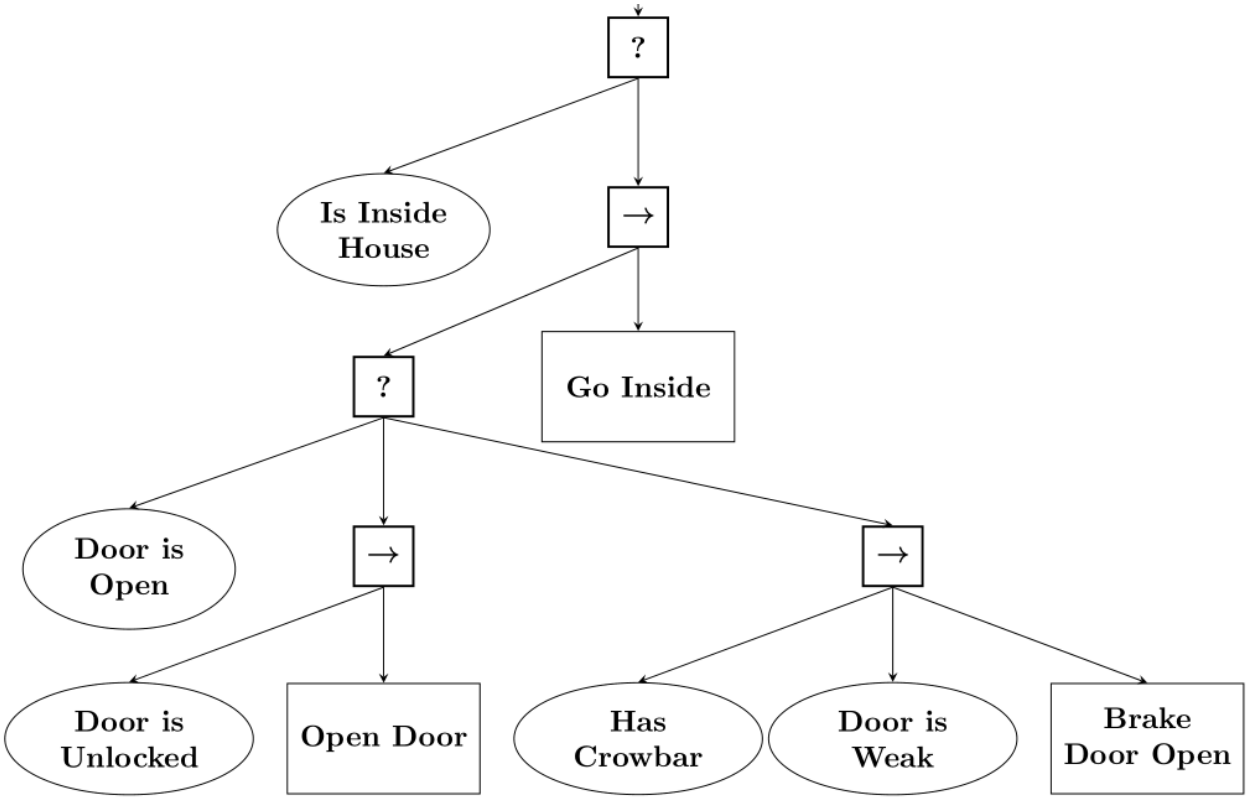


Source: Colledanchise, Ogren: Behavior Trees in Robotics and AI

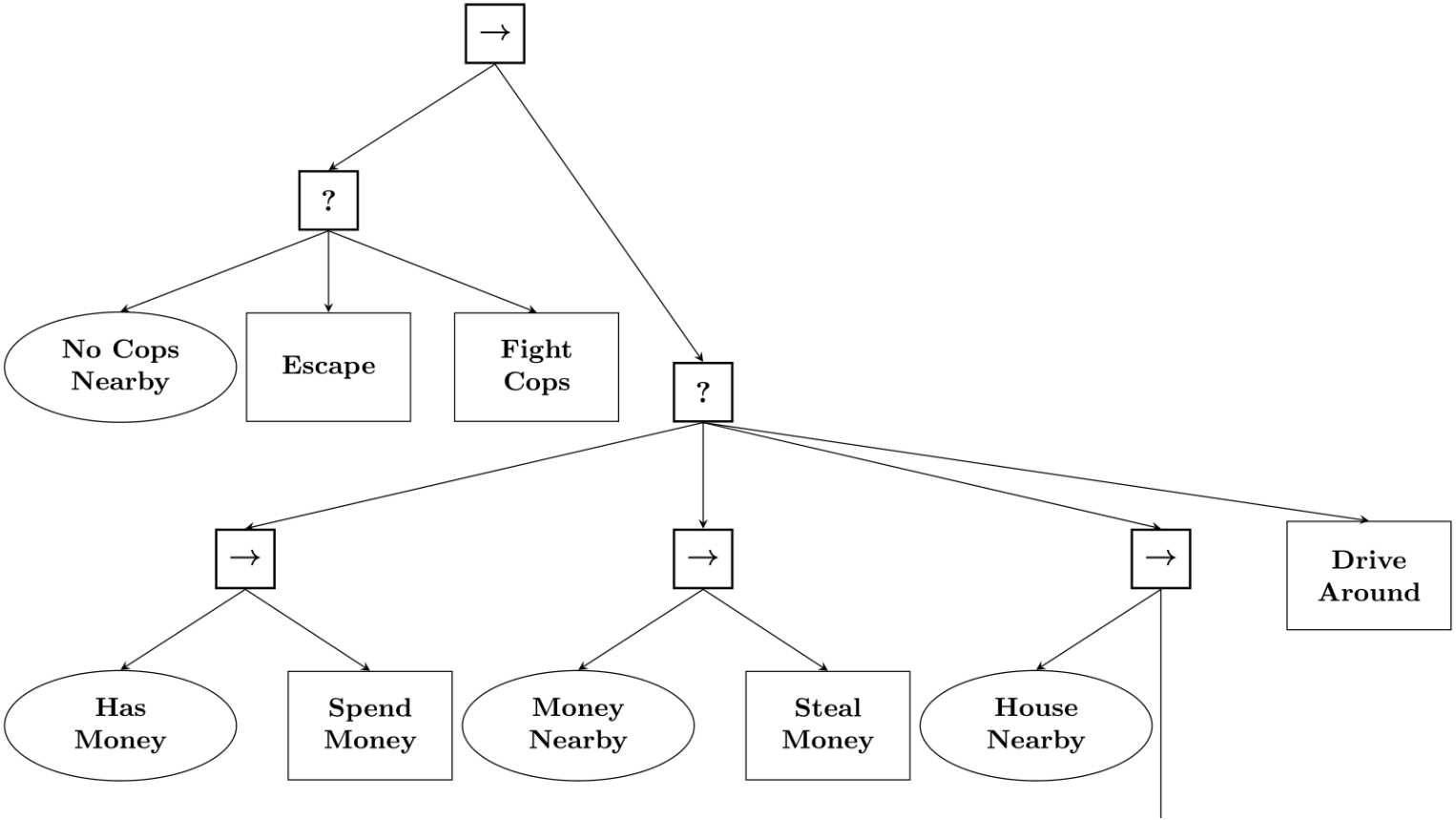
BT – Příklad (Zloděj)



BT – Příklad (Zloděj: Enter House)



BT – Příklad (Zloděj: Escape/Fight)

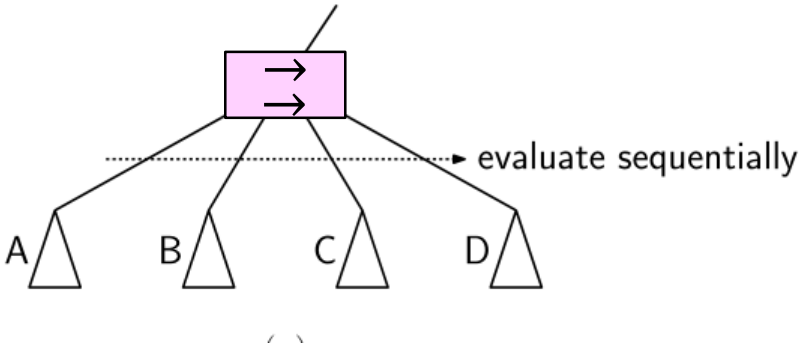


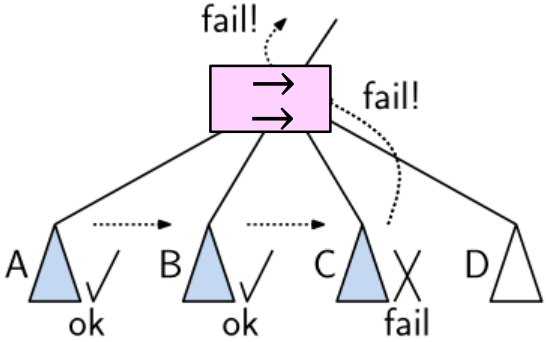
Behavior Trees – Dekorátory

- Pomocné uzly s dodatečnou logikou
  - Parallel sequence
  - Ordered sequence
  - Repeat Nx
  - Until fail
  - Guarding
  - Nedeterministické selektory a sekvence

Behavior Trees – Parallel Sequence

- Vyhodnocuj paralelně dokud neuspěje M uzlů (M <= N)





for *i* ← *1* **to** *N* **do**
  childStatus(i) ← *Tick(child(i))*
if $\sum_{i:childStatus(i)==SUCCESS} \geq M$ **then**
  return *SUCCESS*
else if $\sum_{i:childStatus(i)==SUCCESS} > N - M$ **then**
  return *FAIL*
return *RUNNING*

Behavior Trees - Shrnutí

- Komplexní reaktivní chování reagující na aktuální stav světa
- Využívání předpřipravených modelů
- Skládání komponent

- I. Millington: Artificial Intelligence for Games, kap. 5.4
- Colledanchise, Ogren: Behavior Trees in Robotics and AI https://arxiv.org/pdf/1709.00084.pdf

Obsah přednášky

- Úvod — AIG 1.1 – 1.3
- Rozhodování (decision making) — AIG 5.1 – 5.4
- Plánování cest — AIG 4.1 – 4.3
- Řízení (steering) — AIG 3.2 – 3.4

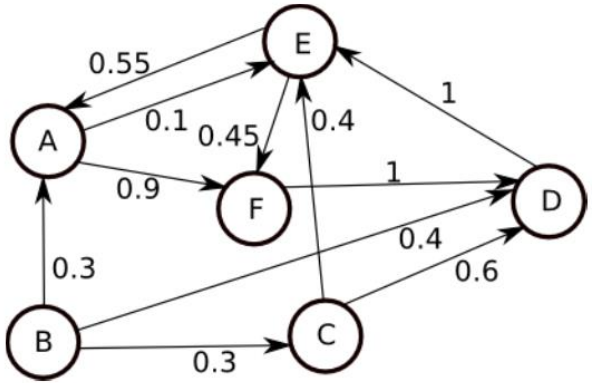
Plánování cest

Oblasti využití

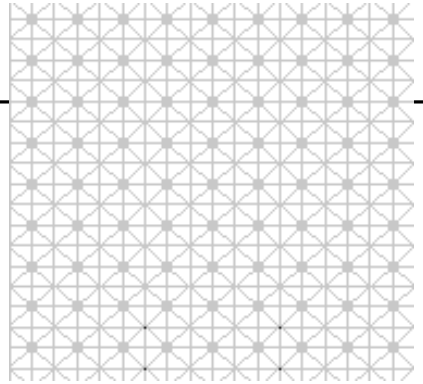
- Podpora pro AI rozhodování (target reachable?)
- Realizace akce / navádění NPC (steering)

Navigační graf

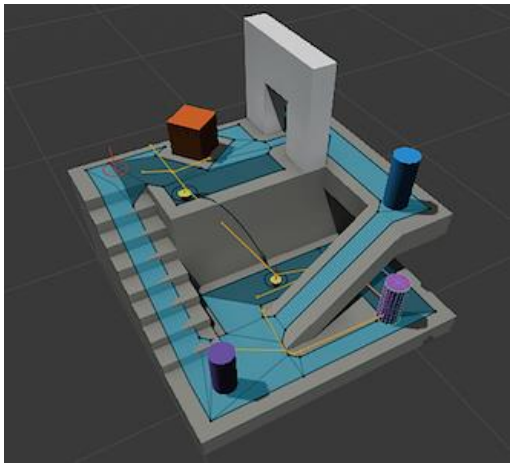
- Orientovaný ohodnocený graf
- Uzly odpovídají místům v prostoru
- Hrany ohodnoceny vzdáleností, případně náročností přechodu z A do B



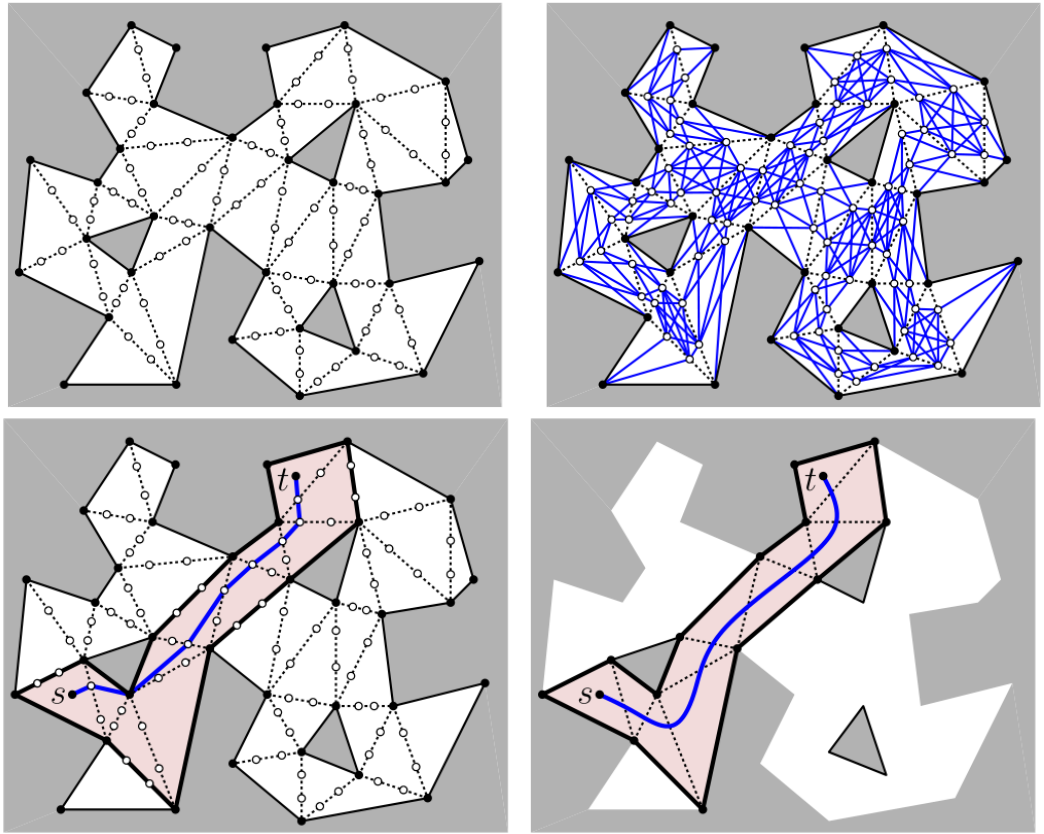
Konstrukce navigačních grafů



- Tiling
  - Uzly pravidelné mřížky
  - Quadtree/Octree
- Body viditelnosti (např. středy místností)
- Expandovaná geometrie
- Navigation Mesh
  - Uzly grafu vrcholy pol. sítě (NavMesh Unity)
  - Polygony ohodnoceny vahou (>=1)
  - Off-mesh links (propojení, skoky)

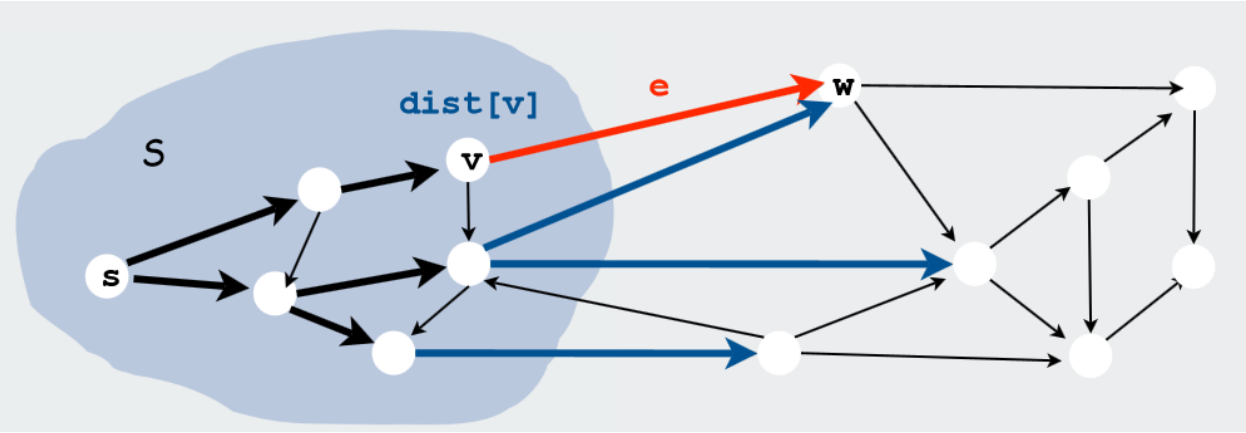


Navigation Mesh - Example



Hledání nejkratší cesty

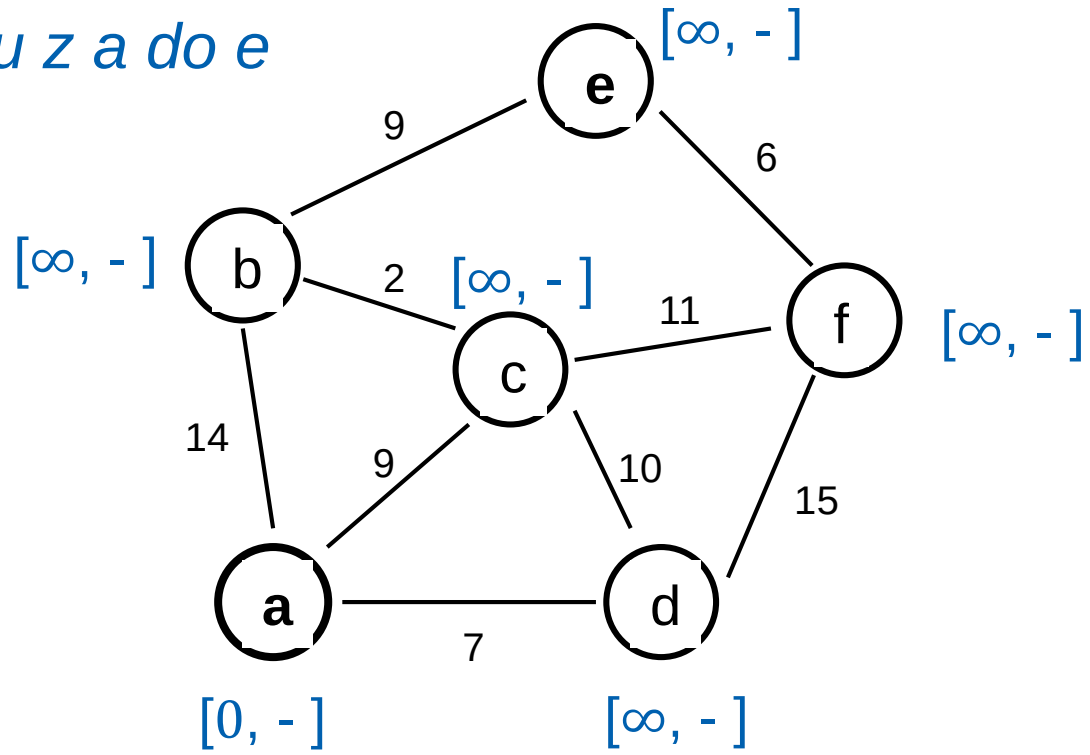
- Dijkstrův algoritmus
  - Strom nejmenších vzdáleností z počátečního uzlu s
  - Nezáporné ohodnocení hran
  - Prohledávání do šířky s prioritní frontou
  - O(|e| + |v| log |v|)



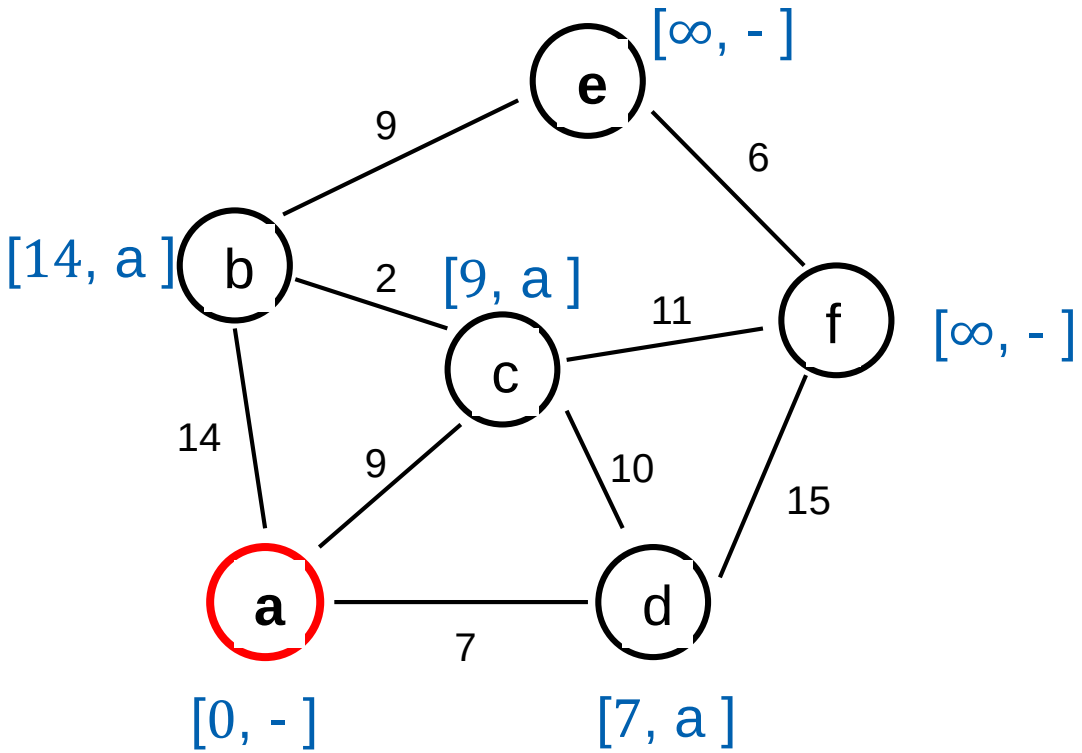
Zdroj: R. Sedgewick

Dijkstrův algoritmus – příklad

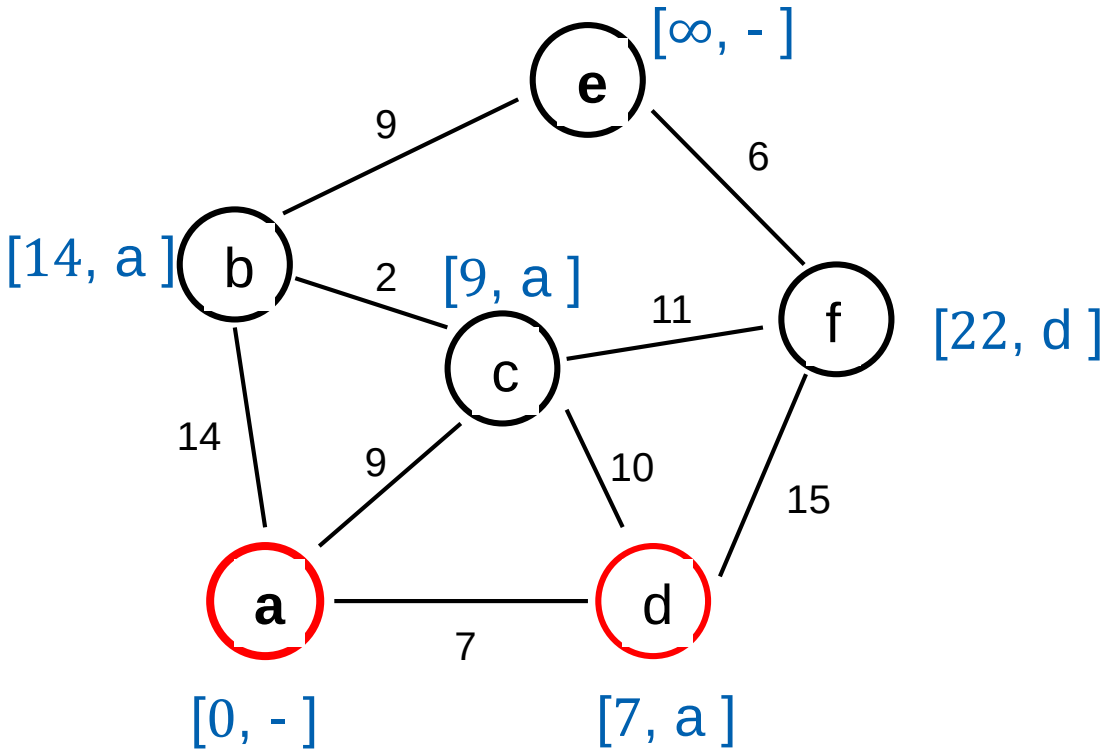
Najdi cestu z a do e



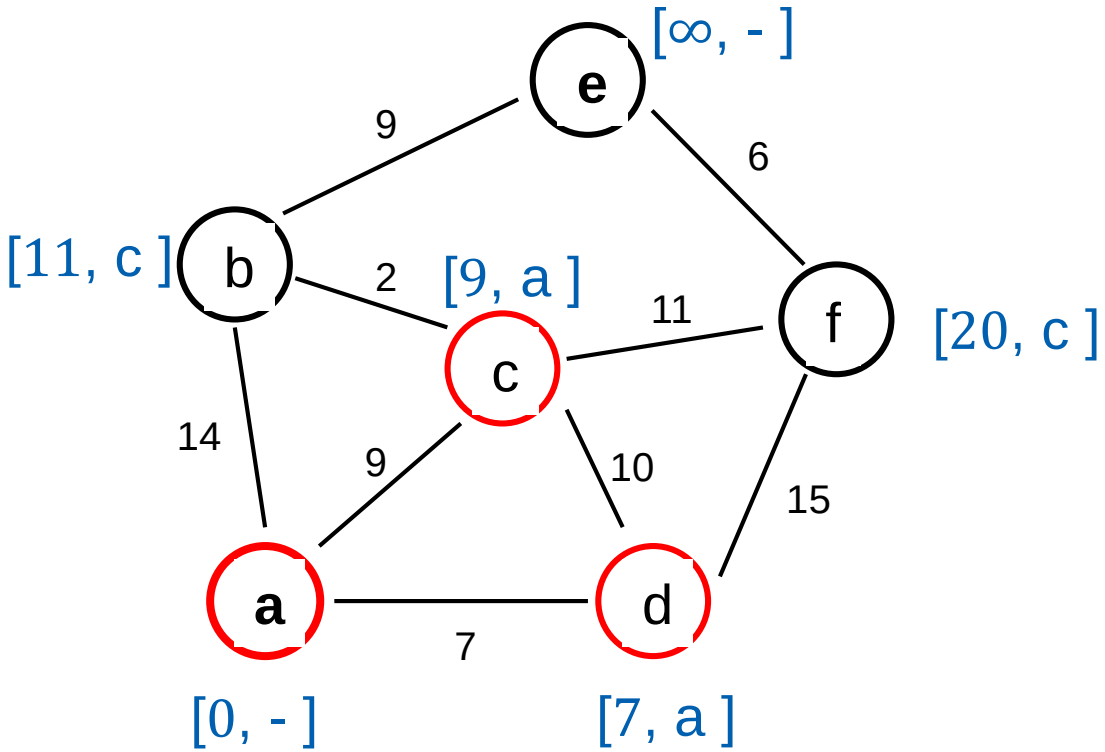
Dijkstrův algoritmus – příklad



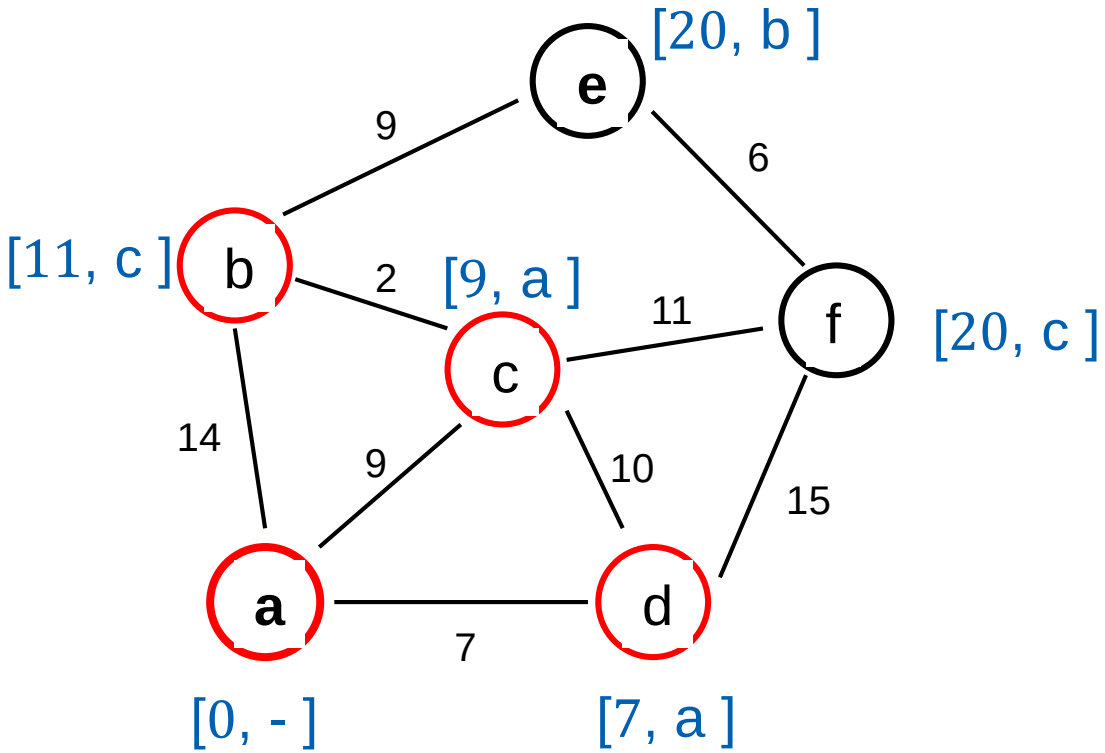
Dijkstrův algoritmus – příklad



Dijkstrův algoritmus – příklad

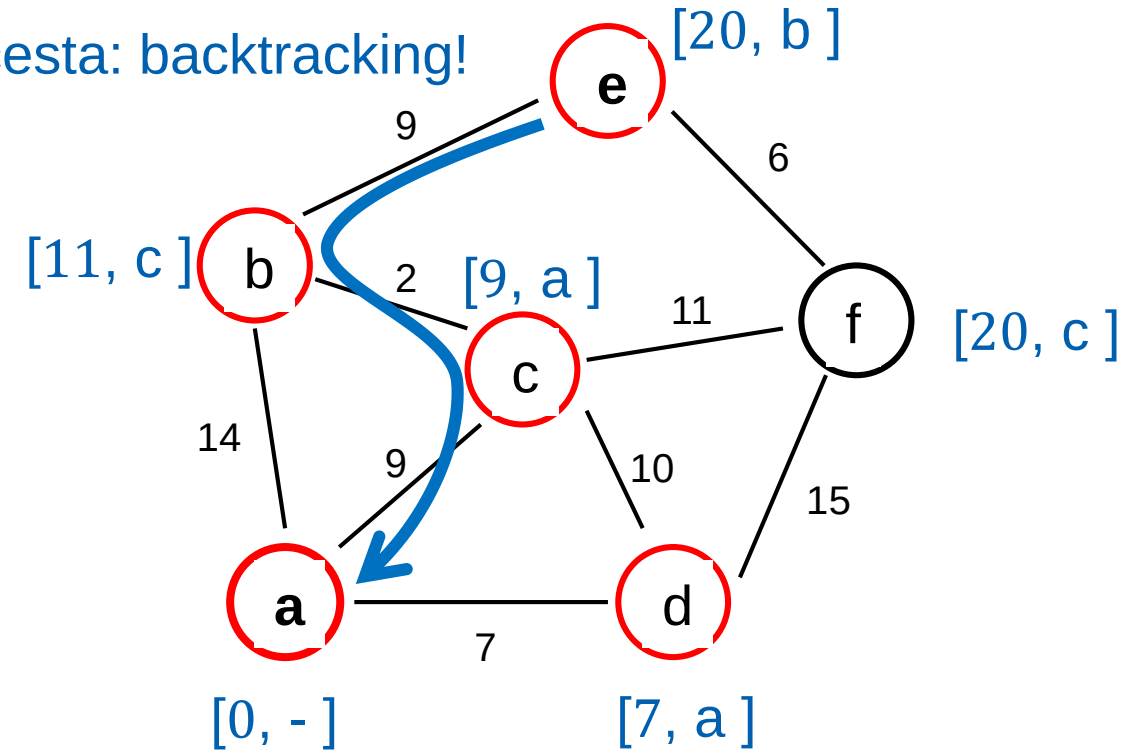


Dijkstrův algoritmus – příklad



Dijkstrův algoritmus – příklad

Nejkratší cesta: backtracking!



A*

- Heuristický odhad vzdálenosti k cíli h(v)
  - Priorita výběru: d(v) + h(v)
- h(v) *podhodnocuje* vzdálenost – přesný výsledek
  - Nejčastěji: Euklidovská vzdálenost od cíle
- Zaměří hledání do smysluplné oblasti
- Většinou výrazně rychlejší než Dijsktrův alg.

- DEMO – Dijkstra & A*
  - Mat Buckland. Programming Game AI by Example. Binaries+Sources: https://www.jblearning.com/catalog/productdetails/9781556220784

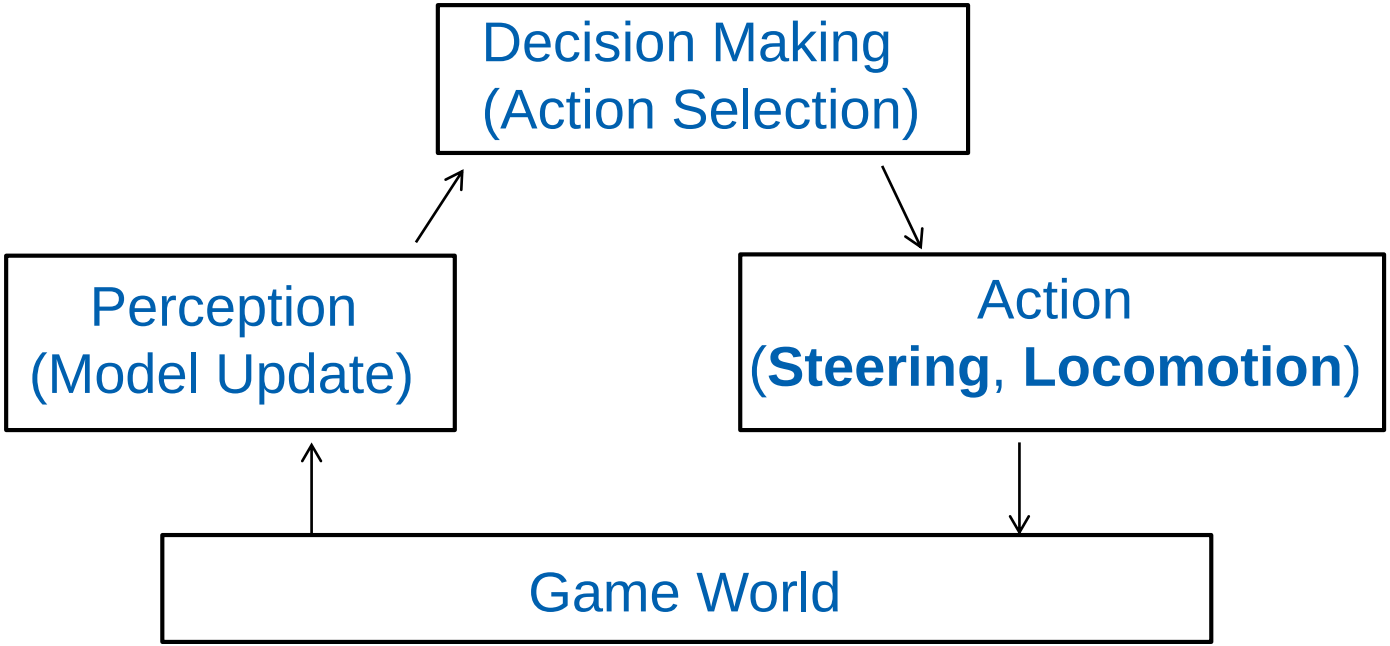
DEMO

- DEMO – Dijkstra & A*
  - Mat Buckland. Programming Game AI by Example. Binaries+Sources: https://www.jblearning.com/catalog/productdetails/9781556220784
- Ohodnoceni hran
  - standardní 26, diagonální 35
  - bahno (x2), voda (x3)

Obsah přednášky

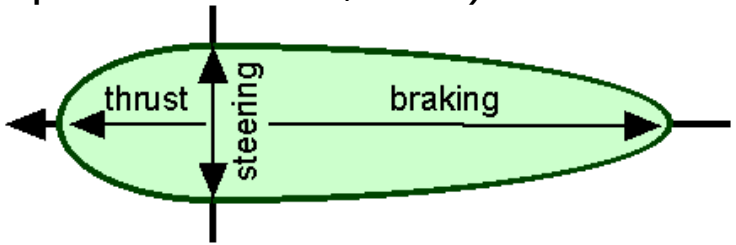
- Úvod — AIG 1.1 – 1.3
- Rozhodování (decision making) — AIG 5.1 – 5.4
- Plánování cest — AIG 4.1 – 4.3
- Řízení (steering) — AIG 3.2 – 3.4

Řízení (Steering)



Steering

- Podle zvolené strategie chování určit
  - Směr pohybu
  - Rychlost pohybu
- Steering: výpočet řídícího vektoru síly (**steering force**)
- Locomotion: aplikace steering force
  - Fyzikální simulace / Animace
  - Mapování na steering force skalární signály (např. řízené auto, NPC)



[Raynolds99] Steering Behaviors For Autonomous Characters

Steering Behaviors – Locomotion

- Jednoduchý locomotion model (vozidla / NPC)

  float mass
  Vector3 position
  Vector3 orientation[3] // basis vectors (forward, up, side)
  Vector3 velocity
  float maxForce
  float maxSpeed

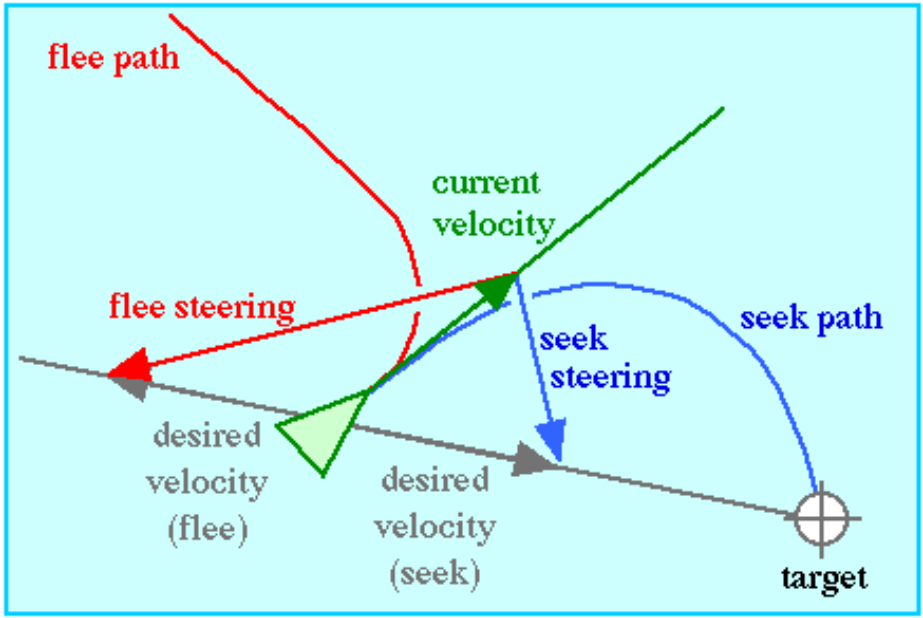
- Locomotion

  steeringForce = truncate(steeringForce, maxForce)
  acceleration = steeringForce / mass
  velocity = truncate(velocity + acceleration, maxSpeed)
  position = position + velocity
  If (alignOrientation)
  Update orientation basis using velocity as forward vector

Hledání a útěk

- Hledání (Seek)

desiredVelocity = normalize(target – position)*maxSpeed
steeringForce = desiredVelocity - velocity



- Útěk (Flee)
  - Opačný steering vektor

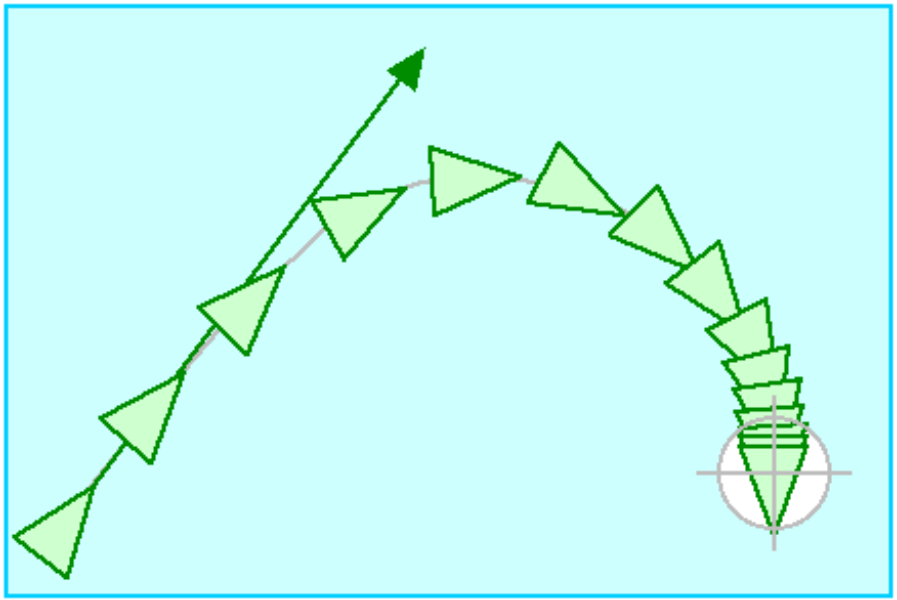
- DEMO

Mat Buckland. Programming Game AI by example. Binaries+Sources: https://www.jblearning.com/catalog/productdetails/9781556220784

Dojíždění

- Dojíždění (Arrive)
- Jako seek, ale zpomaluje

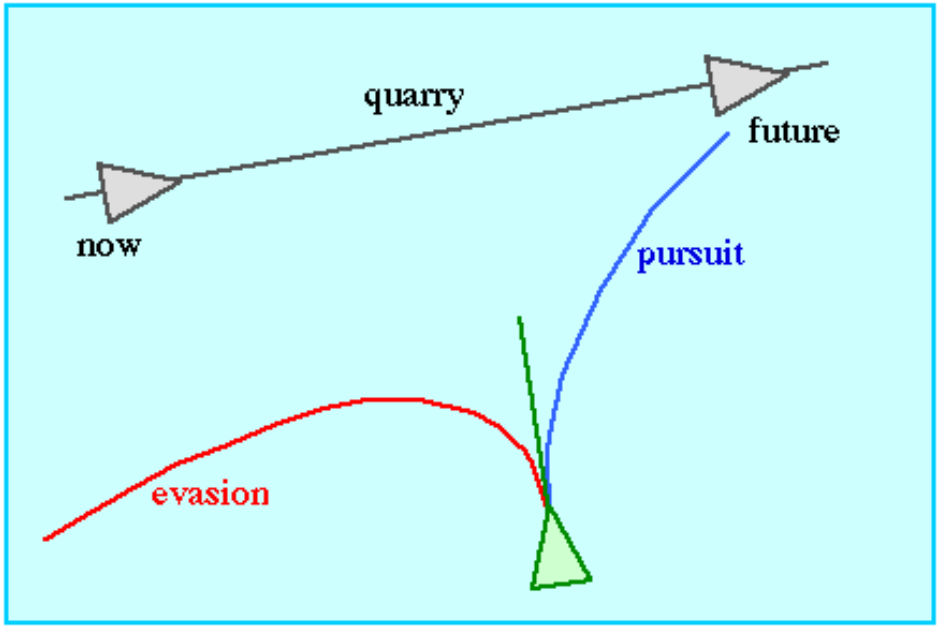
targetOffset = target – position
distance = length(targetOffset)
speed = min(maxSpeed*(distance/slowingDistance), maxSpeed)
desiredVelocity = speed/distance * targetOffset
steeringForce = desiredVelocity - velocity



- Skrývání (Hide)
  - Místo za překážkou + arrive
- DEMO

Pronásledování a vyhýbání

- Pronásledování (Pursuit)
  - Seek na predikovanou pozici
- Vyhýbání (Evade)
  - Flee před predikovanou pozicí

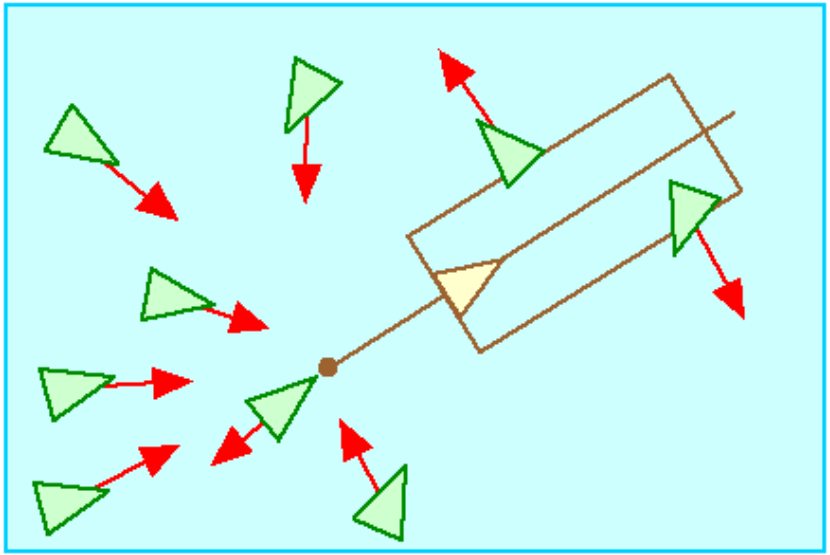


- Mezipozice (Interpose)
  - Centroid dvou pozic pronásledovaných + seek
- DEMO

Následování

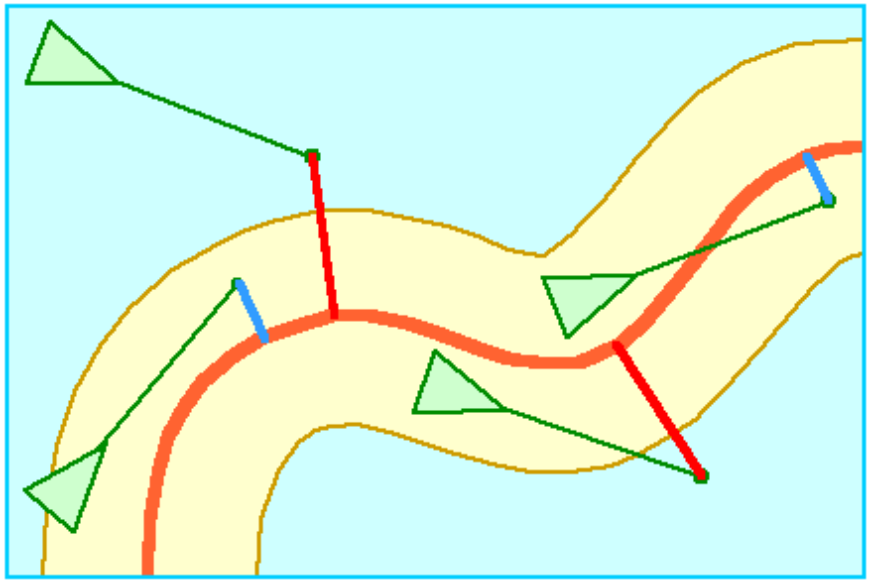
- Následování vůdce

(Leader following / Offset pursuit)



Sledování cesty

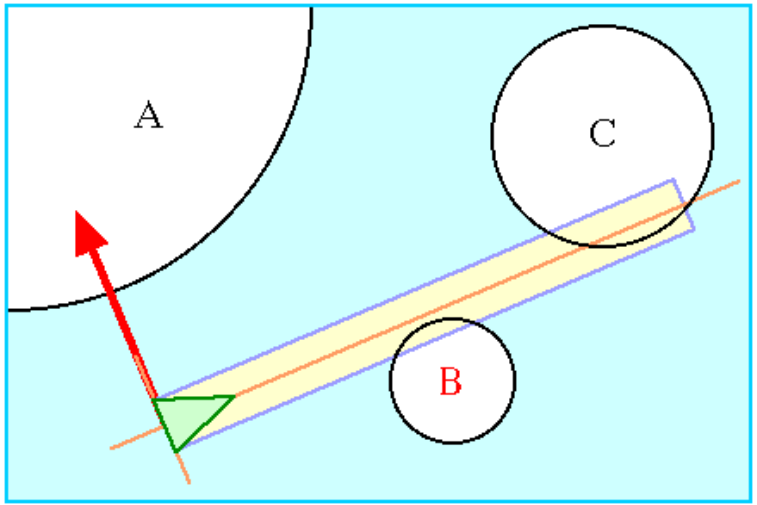
- Sledování cesty (Path following)
- Seek na predikovanou pozici na cestě



- DEMO

Vyhýbání se kolizi

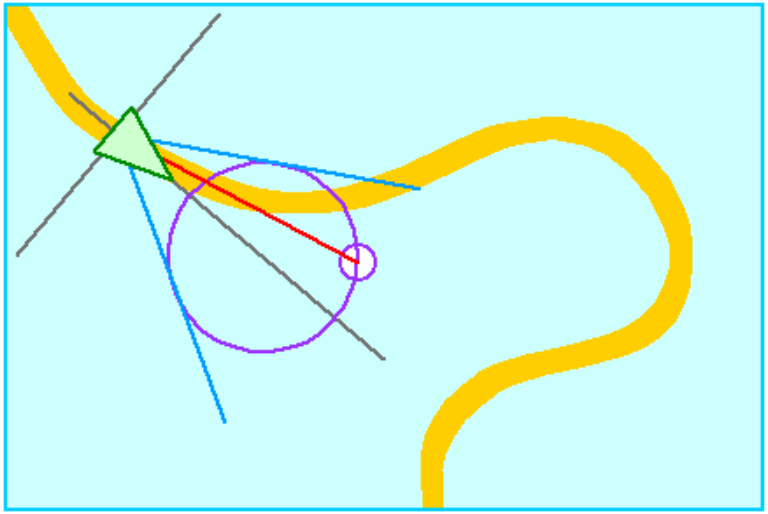
- Vyhýbání se kolizi (Collision avoidance)
  - Zeď
  - Překážka
- Steering force podle nejbližšího kolidujícího objektu



- DEMO

Bloumání

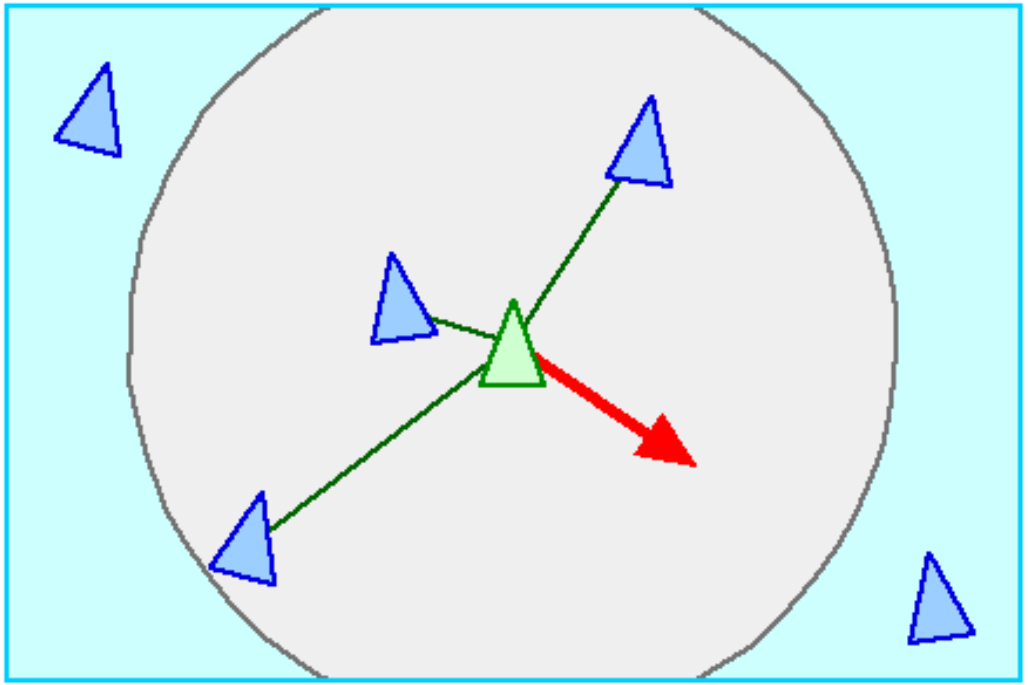
- Bloumání (Wander)



- Náhodný inkrement úhlu na kružnici před NPC
- DEMO

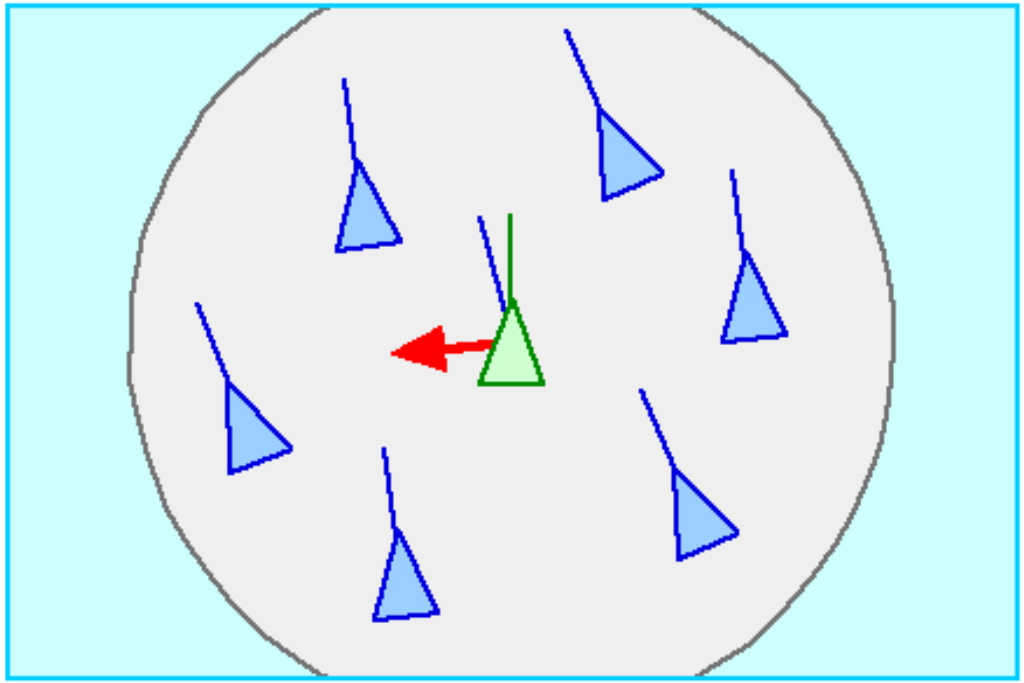
Skupinové chování

- Oddělení (Separation)



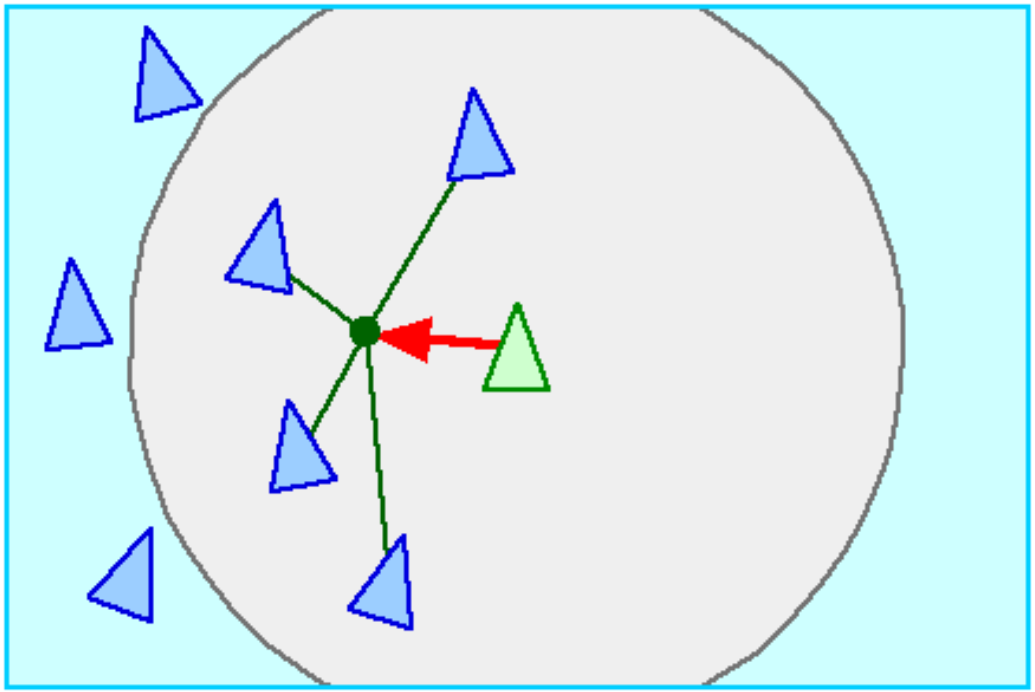
Skupinové chování

- Zarovnání (Alignment)



Skupinové chování

- Soudržnost (Cohesion)



Skupinové chování

- Houfování (Flocking)
  - Vážený součet separation, alignment, cohesion
- DEMO

Obsah přednášky

- Úvod — AIG 1.1 – 1.3
- Rozhodování (decision making) — AIG 5.1 – 5.4
- Plánování cest — AIG 4.1 – 4.3
- Řízení (steering) — AIG 3.2 – 3.4

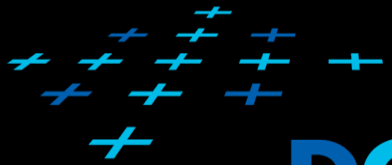
Literatura:

I. Millington, J. Funge. Artificial Intelligence for Games

Mat Buckland. Programming Game AI by Example

D. Mark. Behavioral Mathematics for Game AI

David Mount. Game Programming course notes.



DCGI

KATEDRA POČÍTAČOVÉ GRAFIKY A INTERAKCE

Otázky?