8-bit Games Stike Back: Had

Tomáš Hanzlík

Zadání:

Navrhněte inteligentní řízení hada ve stejnojmenné hře Had. V nejjednodušší variantě jde o čtverečkového hada, který se pohybuje po dvourozměrné mřížce hadovitým stylem, tj. tělo postupně následuje hlavu, a snaží se sníst jedlé čtverečky, které jsou v mřížce různě rozmístěny, případně se mohou objevovat nové. Po pozření jedlého čtverečku se had prodlouží o jeden čtvereček. Musíme ale dávat pozor na to, aby se had nezamotal sám do sebe.

* Hada chápejte jako inteligentního agenta ve virtuálním prostředí, jeho řízení tedy bude realizováno agentní funkcí
* Posudte různé urovně agentních schopností, které se vám mohou hodit: reflexní agent, agent s modelem prostředí, agent s cílem atd.
* Realizujte agentní funkci pomocí probíraných technik, nabízí se: heuristické prohledávání, klasické plánování, různé evoluční techniky

Jako mojí semestrální práci jsem zvolil klasickou arkádovou hru hada. Hada jsem pojal jako izolovaného agenta, který je navíc autonomní a reaktivní.

Task environment:

Jako metrika úspěšnosti byl zvolen počet snězených jablek. Had se pohybuje v prostředí hrací plochy, se kterou nepřetržitě udržuje kontakt, a plánuje tak nejlepší pohyby pro dosažení cílových bodů. Had má neustále přehled o celé hrací ploše a ví, kde se vyskytuje následující bod.

Prostředí:

Jak už bylo řečeno, hrací plocha je pro agenta plně pozorovatelná. Generování nových jablek je stochastické, jablko se náhodně umístí na nějaké volné pole. Pro lehčí implementaci autonomních pohybů jsem zvolil verzi hry, kdy nelze projet skrz zdi.

Implementace chování agenta:

Při průzkumu internetu pro nabrání inspirace, jak je možno řešit tento problém, jsem zpozoroval, že nejúspěšnějších výsledků bylo často dosahováno využíváním modelů strojového učení a neuronových sítí. Tento způsob jsem zvolit nechtěl, neboť toto téma není zcela spojeno s tímto předmětem.

Přístup, který jsem zvolil, se spíše zaměřoval na (ne)informované prohledávání stavového prostoru a úpravu těchto algoritmů. Agent má několik typů chování v závislosti na stavu prostředí.

Na začátku hry hledá had nejkratší cestu k jablku pomocí algoritmu A\*. Tento algoritmus je spustí v každém kroku, a proto je vždy využit jen první plánovaný krok, aby se had s větším tělem zbytečně neobcházel. Toto chování je využíváno do té doby, dokud je splněna následující podmínka: velikost hada < velikost desky / CHANGE\_BEHAVIOUR konstanta. Tuto konstantu jsem zvolil ideální jako číslo 6, které je vhodné pro hrací plochu o velikosti 9x9 a blízké. Pro větší plochy by bylo vhodné konstantu zvětšit.

Pokud podmínka již splněna není, had už nabyl většího těla a musí se nyní přesouvat již nějak systematicky. Jako algoritmus jsem použil též A\* s tím rozdílem, že v prioritní frontě nevybírám dle heuristiky nejbližší pole, ale to nejvzdálenější. Agent sice tedy nevybere nejdelší cestu, ale takovou, aby co nejvýhodněji se uspořádal a systematicky mířil do cíle. Tuto trasu mu stačí naplánovat jednou a poté jí využije celou, neuplatňuje pouze první krok jak v předešlém chování. Agent si zapamatovává, která pole už během jedné cesty za jablkem navštívil, aby se nezacyklil někde v rohu, když vybírá vždy nejvzdálenější vrcholy. A\* byl navíc upraven tak, že si neoznačuje při lokálním prohledávání navštívené vrcholy, neboť by poté nemohl procházet řádek po řádku / sloupec po sloupci, ale pouze ob jeden vynechaný řádek / sloupec. Pohyb by tak neměl hadovitý průběh. Prostředí se dynamicky s každým krokem mění, proto při vkládání pole do prioritní fronty v A\* algoritmu je společně s tím a celkovou vzdáleností přidán též kontejner, ve kterém se vyskytují aktuální pozice částí hada v tomto kroku. Had proto nemusí mít při generování nového jablka přímo k němu přístup, neboť mu jeho tělo může na začátku zavazet, ale než se k němu hlavou dostane, bude už tělo dávno pryč. Pro vyřešení volby nesprávných / slepých cest jsem zvolil taktiku, že v algoritmu A\* při kontrolování sousedů pole vybraného z fronty je zkontrolováno, zda daným pohybem na dané pole bude mít hlava hada přístup k minimálně 70 % všech volných polí (konstanta PERCENTAGE) pomocí algoritmu BFS. Tímto se zamezí potencionální výběr slepé uličky.

Pokud avšak agent nedokáže nalézt cestu k jablku (nejspíše je zablokována jeho tělem a ani brzo se tento stav nezmění), je zde poslední chování vhodně pojmenováno Přežít. Jak už název napovídá, je to pro agenta poslední záchrana a pokus se z pasti dostat. Každé kolo je zvoleno jako následující to pole, které je volné a od jablka nejdále vzdáleno. Pokud by agent poté zjistil, že se mu odkryla cesta k jablku. Přejde zpět na předchozí chování.

Práce na hře mě opravdu bavila. Nemyslím si však, že je autonomní řízení agenta zcela dokonalé. Na hrací ploše 9x9 většinou agent dosáhne půlky zaplněnosti hrací plochy. Stále je co vylepšovat a doufám, že si najdu později čas toto chování ještě vylepšit.