Вештачка интелигенција

Домашна задача 1

Стефан Милев | 206055

1. Децата на Пакман во потрага по богатство

A.

Секоја состојба е претставена на следниов начин:

`(pozicii_na_decata, izminato_vreme)`

`pozicii_na_decata` - торка од `k` торки во формат `(x, y, r)`, каде што `x` е x координатата на секое дете, `y` е y координатата на секое дете, а `r` е ротацијата на секое дете и таа може да прими вредности од 0 до 3: 0 - горе, 1 - десно, 2 - доле, 3 - лево.

`izminato_vreme` - бројот на изминати чекори од почетокот на играта, почнувајќи од 0.

Во проблемот се чува почетната позиција на духот, позициите на препреките и позицијата на богатството.

Со почетната позиција на духот и BFS, се одредуваат позициите на духот во следните чекори и се чуваат во листа, така што `i`-тата позиција на духот се наоѓа на `i`-тата позиција во листата. Со `izminato_vreme` се одредува каде се наоѓа духот во моменталната состојба.

Б.

$$(M \cdot N \cdot 4)^k \cdot t$$

`M, N` - големина на лавиринтот
4 - бројот на можни ротации на децата
`k` - бројот на деца
`t` - изминатото време (позиција на духот)

Првиот (степенуваниот) дел од изразот е добиен од првата торка од состојбата која има облик `(i1, j1, r1), (i2, j2, r2), ..., (ik, jk, rk)`. Домените на променливите се: i - M, j - N, r - 4. Има `k` деца, па затоа се множат толку пати.

Вториот дел од изразот е `t` кој воедно означува позиција на духот и број на изминати чекори во играта. Променливата `t` може да расте до бесконечност, па затоа има бесконечно можни состојби.

В.

$$4^k$$

Има `k` деца и секое од нив има 4 можни акции и во секоја состојба сите деца превземаат некоја акција.

Г.

Почетна состојба:

```
(((2, 0, 3), (0, 3, 2), (3, 5, 0)), 0)
```

Целна состојба:

Една од торките во `pozicii_na_decata` е еднаква на позицијата на богатството.

Д.

Во секоја акција, сите деца превземаат по една од нивните акции (заврти се лево, заврти се десно, придвижи се напред, стоп).

Легални акции во секоја состоба се тие во кои:

- ни едно дете не стапнува на препрека или духот
- ни едно дете не излегува надвор од лавиринтот
- ни едно дете не стапнува врз друго дете

Python код:

```
def successor(self, state):
    next_states = {}
    children = state[0]
    time = state[1]
    # сите можни акции
    combinations =
itertools.combinations_with_replacement(self.possible_actions.keys(), 5)
    # итерирање на сите можни следни акции
    for combination in combinations:
        new_children = []
        # составување на акција без разлика дали е легална
        for action, child in zip(combination, children):
            # ако акцијата е stop, не прави ништо
            if action == 'stop':
                new_children.append(child)
            # ако акцијата е go_forward, одреди ја следната позиција
            elif action == 'go_forward':
                child = (child[0] + self.moves[child[2]][0], child[1] +
self.moves[child[2]][1], child[2])
                new_children.append(child)
            # ако акцијата е turn_left или turn_right, одреди ја следната
насока
            else:
                direction = (child[2] + self.possible_actions[action]) % 4
                child = (child[0], child[1], direction)
                new_children.append(child)
        # отфрли акција ако дете стапне на препрека
        if any(child[:2] in self.walls for child in new_children):
            continue
        # отфрли акција ако дете стапне на дух
        if any(child[:2] == self.ghost_positions[time] for child in
new_children):
```

```
            continue
        # отфрли акција ако дете излезе од лавиринтот
        if min(c[0] for c in new_children) < 0 or max(c[0] for c in
new_children) >= self.m or min(c[1] for c in new_children) < 0 or max(c[1]
for c in new_children) >= self.n:
            continue
        # отфрли акција ако едно дете стапне на друго
        if len(set(child[:2] for child in new_children)) !=
len(new_children):
            continue
        # акцијата е легална
        next_states[str(combination)] = (tuple(new_children), time + 1)
    return next_states
```

Ѓ.

Нетривијална допустлива евристика не е возможно да се дефинира затоа што децата не знаат каде се наоѓа богатството.

Тривијална допустлива евристика:

Евристиката `h(n) = 0`. Допустлива е затоа што е помала од вистинското растојание до секоја цел. Евристиката во секое теме, како и во целната состојба, е 0.

Е.

Минималното растојание меѓу децата и колоната `K`. Оваа евристика е допустлива затоа што вистинското растојание до богатството секогаш ќе биде поголемо или еднакво од минималното растојание на децата до колоната `K`, така што најблиското дете би се движело само лево или само десно.

Ж.

DFS:

Не може да се реши проблемот со DFS затоа што децата ќе вртат кругови бесконечно и нема да го најде оптималното решение.

UCS:

Може да се реши проблемот со UCS. Сите акции имаат иста цена (цената е 1) и затоа UCS ќе се однесува исто како BFS, така што ќе го најде оптималното решение.

BFS:

Може да се реши проблемот со BFS и ќе го најде оптималното решение. BFS ќе се однесува исто како UCS.

A*:

Може да се реши проблемот со A*. Доколку евристиката е тривијална допустлива, тогаш ќе се однесува исто како UCS и затоа што цените се исти, ќе се однесува исто како и BFS. Ако децата знаат дека богатството се наоѓа во колоната K, тогаш со евристиката „Растојанието на најблиското дете до колоната“ ќе може да се најде оптималното решение во помал број на чекори од другите три алгоритми. Ќе го најде оптималното решение секогаш доколку евристиката е допустлива.

Проблемот може најбрзо (во најмал број чекори) да се реши со A* доколку децата знаат во која колона се наоѓа богатството, но доколку не знаат, A*, BFS, UCS ќе се однесуваат исто и ќе го најдат решението во ист број на чекори, користејќи тривијална допустлива евристика за A* пребарување.

2. Дипломатска вечера

A.

Променливи:

- VIP1
- VIP2
- VIP3
- VIP4

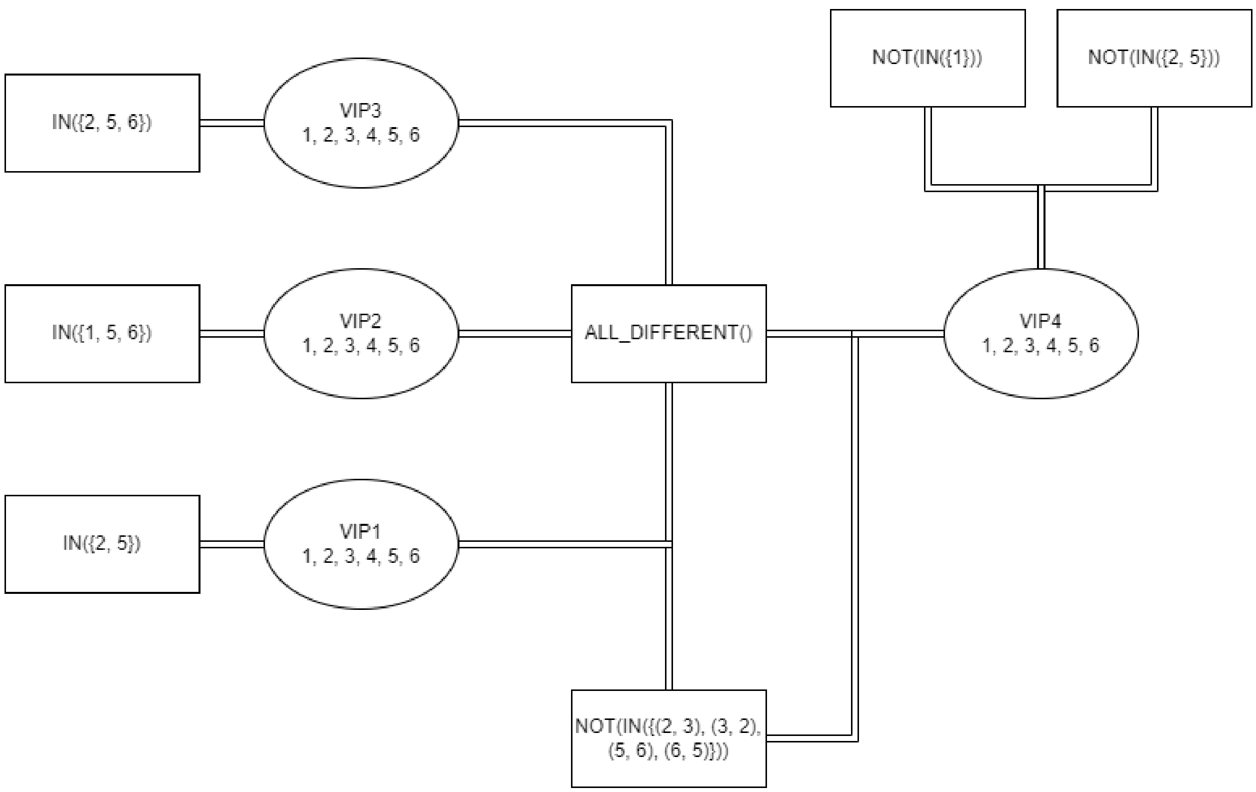
Домен:

- 1
- 2
- 3
- 4
- 5
- 6

Услови:

1. VIP1 ≠ VIP2 ≠ VIP3 ≠ VIP4
2. VIP3 ∈ {2, 5, 6}
3. VIP1 ∈ {2, 5}
4. (VIP1, VIP4) ∉ {(2, 3), (3, 2), (5, 6), (6, 5)}
5. VIP2 ∈ {1, 5, 6}
6. VIP4 ∉ {1}
7. VIP4 ∉ {2, 5}

Б.



В.

Чекор	VIP1	VIP2	VIP3	VIP4	Услов
0	1, 2, 3, 4, 5, 6	1, 2, 3, 4, 5, 6	1, 2, 3, 4, 5, 6	1, 2, 3, 4, 5, 6	-
1	2, 5	1, 2, 3, 4, 5, 6	1, 2, 3, 4, 5, 6	1, 2, 3, 4, 5, 6	3.
2	2, 5	1, 5, 6	1, 2, 3, 4, 5, 6	1, 2, 3, 4, 5, 6	5.
3	2, 5	1, 5, 6	2, 5, 6	1, 2, 3, 4, 5, 6	2.
4	2, 5	1, 5, 6	2, 5, 6	2, 3, 4, 5, 6	6.
5	2, 5	1, 5, 6	2, 5, 6	3, 4, 6	7.

Г.

Реброто (VIP1, VIP4) е конзистентно затоа што се можни паровите (2, 4), (2, 6), (5, 3), (5, 4), додека паровите (2, 3) и (5, 6) не се дозволени.

Реброто (VIP4, VIP1) е конзистентно затоа што се можни паровите (3, 5), (4, 2), (4, 5), (6, 2), додека паровите (3, 2) и (6, 5) не се дозволени.

Ребрата на првиот услов се конзистентни затоа што нема променливи со исти дозволени вредности после исполнување на унарните услови и може сите променливи да добијат различни вредности.

Сите други ребра се конзистентни, па затоа и целиот граф е конзистентен.

Д.

Услови:

8. VIP1 ≠ VIP2 ≠ VIP3 ≠ VIP4
9. VIP3 ∈ {2, 5, 6}
10. VIP1 ∈ {2, 5}
11. (VIP1, VIP4) ∉ {(2, 3), (3, 2), (5, 6), (6, 5)}
12. VIP2 ∈ {1, 5, 6}
13. VIP4 ∉ {1}
14. VIP4 ∉ {2, 5}

Доделување на вредности

Чекор	VIP1	VIP2	VIP3	VIP4
1	**2**	-	-	-
2	2	-	**5**	-
3	2	**1**	5	-
4	2	1	5	**4**

Останати можни вредности (forward checking)

Чекор	VIP1	VIP2	VIP3	VIP4
1	-	1, 5, 6	5, 6	4, 6
2	-	1, 6	-	4, 6
3	-	-	-	4, 6
4	-	-	-	-

Користената евристика во сите чекори е MRV - Minimum Remaining Values. Доколку има променливи со ист број на можни вредности, земена е променливата што доаѓа прва лексикографски.

Решение:

- VIP1: 2
- VIP2: 1
- VIP3: 5
- VIP4: 4