Ćwiczenia 3 SI

Maurycy Borkowski

12.05.2020

1*

Algorytmy mrówkowe to technika szukania dróg w grafie bazująca na metodzie mrówek szukających pożywienia. Mrówki szukają optymalnej drogi do pożywienia idąc z mrowiska wypuszczają feromony, które z czasem praują. Mrówki poruszają się w miarę losowo lub po śladzie feromonów im mocniejsze natężenie tym więcej mrówek tam pójdzie. Feromony parują po jakimś czasie więc dalekie (nie optymalne) ścieżki nie będą eksplorowane, a optymalne często chodzone będą częściej, aż w końcu wszystkie mrówki będą chodziły po jednej optymalnej ścieżce.

Aplikacja w problemie komiwojażera:
Wypuszczamy mrówki przekazując im zasady:

- Każde miasto odwiedzamy raz.
- Bliższa odległość = większe prawdopobieństwo wybrania.
- Natężenie feromonów większe = większe prawdopobieństwo wybrania.
- Po sukcesie czyli objechaniu wszystkich miast naznaczamy naszą trasę feromonami odpowiednio przeskalowane jak długa była nasza trasa.

2

Zmienne:
poszczególne pola z dziedznami (0..1)
bloki (ich długości) z dziedzinami (0..N/M)
początki bloków z dziedzinami (0..N/M)

Więzy:
$blok\_x\_i\# = dlugosc \quad z \quad inputu$
$blok\_y\_j\# = dlugosc \quad z \quad inputu$

$blok\_x\_i\# = k \quad and \quad pocz\_blok_{x,0}\# = y \quad \# ==> (pole_{x,y} + pole_{x,y+1} + \cdots +$

$pole_{x,y+k} \quad \# = \quad k$
Podobnie z y.

$pocz\_blok_{x,i+1} \# > pocz\_blok_{x,i} + blok\_x, i$

Dla *sensownych* dowolnych x,y,i,j,k

3

Tworzymy zmienne z dziedzinami:
$zajecia_i \quad in \quad (1 \ldots 50) \qquad dla \quad i \in [0, N]$
$prowadzacy_i \quad in \quad (0 \ldots M) \qquad dla \quad i \in [0, N]$
$grupa_i \quad in \quad (0 \ldots K) \qquad dla \quad i \in [0, N]$
N to liczba różnych zajęć do rozłożenia.

Tworzymy więzy:
$prowadzacy_i \quad \# = \quad$ *odpowieni prowadzący dla i-tych zajęć*
$grupa_i \quad \# = \quad$ *odpowienia grupa dla i-tych zajęć*

Węzły eliminujące okienka (dla liczb 2-9):
PRZYKŁAD: 2
$zaj_i = zaj_{i+2} \implies zaj_{i+1} = zaj_i$

$(grupa_i \# = grupa_j) \quad and \quad abs(zajecia_i - zajecia_j) \# = 2 \quad \# ==> OR(\quad ((grupa_k \# = grupa_i) \quad and \quad zajecia_k \# = \quad min(zajecia_i, zajecia_j) + 1))$ dla wszystkich k
Jeszcze trzeba zadbać o to by możliwe były zajęcia do 19 i od 11 następnego dnia. Można to zrobić dodając kolejną koniunkcję na to że oba zajęcia muszą być jednego dnia w implikacji.

Węzły eliminujące te same zajęcia grupy i prowadzących w jednym momencie:
$(zajecia_i \# = zajecia_j) \quad \# ==> \quad (grupa_i \#! = grupa_j) and (prowadzacy_i \#! = prowadzacy_j)$

4

Funkcja powinna uwzględniać (minimalizujemy wartość):

- Obowiązki nie mogą się pokrywać!
- Okienko jest nie fajne. Dodajemy sumę kwadratów długości wszystkich okienek.

- Dzień wolny jest fajny. Dzień wolny to np. -10 a poniedziałek lub piątek -20
- Kończenie lub rozpoczynanie o skrajnych godzinach +5
- Naładowanie dużo do jednego dnia jest złe. Powyżej 8h każda godzina to +3 punktów
- Wedle preferencji można premiować jeżeli zachowana jest kolejność ćwiczenia → wykład lub odwrotnie.

5,6

Pentago

Będę wyróżniał i punktował odpowiednio układy (w jednym kwadracie):

- 3 kulki w linii — 5 punktów
- 3 kulki w linii ukośnej — 3 punkty
- 2 kulki w linii — 3 punkty
- 2 kulki w linii ukośnej — 2 punkty
- Kulka w środku — 1 punkt

Lis i gęsi

Funkcja którą wilk chcę maxymalizować a gęsi minimalizować:
liczba możliwych ruchów wilka (bicie robimy *ile wlezie*) + (1 - liczba_gęsi)

7

Rzeczywiście gracz numer 2 ma dużo trudniejsze zadanie. Gracz z numer 1 rozpoczyna środkiem, gracz numer 2 też powinien sprawdzić to pole, ale prawdopodbnie będzie zajęte. To mu daje najwięcej możliwości skończenia gry z jego nieprzegraniem. Grając czysto losowo po takim rozpoczęciu gracz numer dwa ma szanse na nieprzegranie w ciągu dwóch swoich ruchów:

$$P = \frac{1}{7} + \frac{1}{6} = \frac{13}{42} \approx 0,31$$

Może wnioskować, ale to i tak nie ratuję jego sytuacji mocno. (Nie blokuje tej samej linii, jak trafi w 1. to blokuje mu linię)
Pierwszy strzał blokuje dowolną linię, jeżeli nie zostanie zablokowana to drugi strzał blokuję inną linię (linie przechodzące przez środek).
Niesety żadna strategia nie pomoże mu osiągnąć dużo lepszego startu. Później strategią drugiego powinno być strzelanie żeby blokować by doprowadzić do remisu. Pierwszego wybieranie losowej linii i uzupełnianie jej. Jedyną szansą na wygranie 2. jest gdy 1. da w drugiej turze do rogu dlatego 1. powinien najpierw próbować dawać nie do rogu.
Buczenie wykorzystujemy oczywiście do wnioskowania, której lini nie opłaca się dokładać lub którą linię tym bardziej opłaca się blokować z perespektywy gracza numer 2.

8

Rozważmy wierzchołki w porządku odwrotnym do porządku topologicznego.
W tym DAG-u jak rozpatrujemy jakiś wierzchołek V to wszystkich jego synów już rozpatrzyliśmy, więc znamy wartości $V_{opt}(u)$ dla każdego u syna V. Rozpatrując więc V bierzemy po prostu max z każdej akcji, która doprowadza nas do jednego z naszych synów. Tak po jednej symulacji mamy optymalną politykę.

$O(V + E)$

10

a

W MCTS czas poświęcony na znalezienie poprzedniego ruchu wykorzystujemy korzystając z danych zebranych do wykonania bieżącego ruchu (zachowujemy wartości w wierzchołkach).

W Alpha-Beta-Search wykorzystujemy tak samo jak w MCTS zachowujemy dane z poddrzewa do którego idziemy.

b

W MCTS możemy przeprowadzać symulacje na synach rozwijanego węzła równolegle.

W Alpha-Beta-Search możemy dla na N wątkach puścić funkcję na każdym synie lub synie syna etc. do wyszukania MIN/MAX w danym poddrzewie.