Sztuczna inteligencja
Ćwiczenia 3
Pierwsze zajęcia w maju

Każde zadanie warte jest 1 punkt. Zadanie z gwiazdką nie wlicza się do maksimum.

Zadanie 1. W zadaniu rozważymy ogólną metodę binaryzacji, dla więzów, które są zdefiniowane za pomocą wyrażeń arytmetycznych i symboli relacyjnych (przykładowo $2A+4B > 7C+D^2+EF+G^3$). Pokaż, jak zamieniać takie więzy na więzy o arności 2 lub 3 (być może dodając nowe zmienne, pamiętaj o tym, że nowe zmienne muszą mieć dziedziny). A następnie pokaż, jak eliminować więzy o arności 3 (zamieniając je na binarne).

Zadanie 2. Przypomnij, co to jest spójność łukowa i algorytm AC-3. Osiąganie spójności łukowej może być kosztowne, zwłaszcza, gdy dziedziny zmiennych są duże (dlaczego?). Można tę spójność przybliżać, zajmując się jedynie krańcami dziedzin (czyli wartością najmniejszą i największą). Powiedzmy, że więzy mają postać: $\sum_{i=0}^{N} c_i x_i \circ y$, gdzie c_i są stałymi, x_i, y to zmienne FD, a $\circ \in \{<, \leq, =\}$. Opisz algorytm wnioskowania (propagacji), który analizując kolejne więzy stara się w możliwie największym stopniu ograniczać ich dziedziny (ale zmieniając jedynie dolne i górne ograniczenia tych dziedzin). Oszacuj jego złożoność (czasową i pamięciową).

Zadanie 3. (1p$\star$) Przeczytaj i opowiedz o tym, czym sa algorytmy mrówkowe (ant colony algorithms). Powiedz, jak zastosować je do rozwiązania problemu komiwojażera.

Zadanie 4. (1p) Opisz poniższe algorytmy (możesz użyć nazw, jeżeli znasz):

a) local beam search dla $k = 1$,

b) local beam search z jednym początkowym stanem i bez limitu na liczbę zachowanych stanów po generacji następnika,

c) symulowane wyżarzanie z $T = 0$ przez cały czas działania algorytmu,

d) symulowane wyżarzanie z $T = \infty$ przez cały czas działania algorytmu,

e) algorytm genetyczny z populacją wielkości 1.

Odpowiedzi (w razie potrzeby) uzasadnij.

Zadanie 5. (1p) Podobnie jak burze, obrazki logiczne można również rozwiązać za pomocą więzów w Prologu. Opisz, jak modelować obrazki logiczne używając więzów dostępnych w SWI-Prologu (zobacz ostatnie slajdy W4 i początek wykładu W5, możesz też skonsultować się z dokumentacją modułu clpfd w SWI-Prologu).

Zadanie 6. (1p) Rozważamy uproszczone zadanie układania planu lekcji, w którym (między innymi) nie przejmujemy się dostępnością sal.

- Mamy pewną liczbę zajęć do rozmieszczenia.
- Każde zajęcia mają następujące parametry: klasa (w znaczeniu „grupa uczniów") oraz nauczyciel.
- Każdemu zajęciu należy przypisać termin: liczbę od 1 do 50. Liczby od 1 do 10 oznaczają poniedziałek, od 11 do 20 – wtorek, itd.
- Żadna klasa nie może mieć „okienka", czyli przerwy w zajęciach. Czyli każdego dnia przychodzi na jakąś godzinę, ma zajęte wszystkie kolejne godziny, a potem idzie do domu i do szkoły przychodzi następnego dnia.

Przedstaw to zadanie jako problem więzowy, który da się wyrazić za pomocą konstrukcji dostępnych w SWI-Prologu. Przy definiowaniu warunków na bezokienkowość planu możesz potrzebować dodatkowych zmiennych.

Zadanie 7. (1p) Przedstaw zadanie układania planu zajęć w Instytucie Informatyki jako problem spełnialności więzów, w którym oprócz twardych wymagań, określających jakie plany są dopuszczalne[1], istnieje funkcja określająca jakość dopuszczalnego planu i tym samym jesteśmy zainteresowani znalezieniem dopuszczalnego planu maksymalizującego tę jakość. Najważniejszą częścią tego zadania jest przedstawienie propozycji takiej funkcji, czyli próba precyzyjnego opisania, co sprawia, że jakiś plan jest lepszy od innego. Powinieneś uwzględnić wyniki głosowania. Możesz też zaproponować jakąś procedurę zdobycia dodatkowej wiedzy na temat preferencji studentów, jeżeli uznasz, że samo głosowanie to za mało.

Zadanie 8. (1p), ⋆ To zadanie jest z gwiazdką, bo dotyczy wiedzy 'nieobowiązkowej'. Adresowane jest do osób, które znają Prolog.

a) Predykat `labeling` można napisać korzystając z bardziej elementernych predykatów. Twoim zadaniem jest napisanie najprostszej wersji, w której zmienne obsługiwane są od lewej strony, w której nie przejmujemy się kolejnością wyboru wartości. Możesz używać predykatu `indomain/1`.

b) Jak samodzielnie zdefiniować predykat `indomain`, korzystając z predykatu `fd_dom(+Var, -Dom)` odczytującego dziedzinę zmiennej[2].

c) W programach prologowych, które powstawały do tej pory, `labeling` był na końcu głównego predykatu rozwiązującego. Zastanów się, jak wpłynęłyby na poprawność i efektywność procedury wyszukiwania rozwiązania inne pozycje tego predykatu.

Zadanie 9. (1p) Heurystyczne algorytmy można łączyć ze sobą w dość dowolne kombinacje. Zaproponuj sensowne[3] połączenie:

a) algorytmów ewolucyjnych i hill climbing,

b) A^* oraz local beam search,

c) symulowanego wyżarzania i algorytmów ewolucyjnych (inaczej niż w podpunkcie a, z symulowanego wyżarzania powinniśmy wziąć jedynie ogólną ideę),

d) taboo search z algorytmami ewolucyjnymi.

W razie potrzeby krótko wyjaśnij, jak działają oryginalne algorytmy.

Zadanie 10. (1p) Wybierz jedną z następujących gier: lis i gęsi, breakthrough (board game), pentago, skoczki (gra planszowa). Zaproponuj dla wybranej gry heurystyczną funkcję oceniającą sytuację na planszy. Wyraźnie wskaż, jakie ta funcja ma parametry.

Uwaga: zapytanie: *<gra-X> wikipedia* zawsze prowadzi do odpowiedniego opisu. Bądź przygotowany na krótkie opisanie zasad gry (w razie potrzeby).

Zadanie 11. (1p), ⋆ Jak wyżej, tylko inna gra.

[1] W znacznym stopniu analogicznych do przedstawionych w poprzednim zadaniu.

[2] W dokumentacji jest zasugerowane, jak to zrobić, ale z użyciem mechanizmu DCG. W Twoim rozwiązaniu nie powinieneś z niego korzystać.

[3] Trudno zdefiniować precyzyjnie sensowność. Na potrzeby tego zadania przyjmiemy, że sensowne oznacza przekonanie autora, że warto takie połączenie sprawdzić w pewnych zadaniach – i że może ono zadziałać lepiej niż każdy ze składników kombinacji