Laboratorium BO

Opis posługiwania się pakietem AMPL w zakresie przydatnym do wykonania ćwiczeń:
Programowanie liniowe i Programowanie całkowitoliczbowe

Pakiet AMPL

Pakiet AMPL jest narzędziem do rozwiązywania liniowych, nieliniowych i całkowitoliczbowych zadań programowania matematycznego. W jego skład wchodzą: algebraiczny język modelowania, różnorodne solwery służące do rozwiązywania modeli programowania matematycznego oraz okienkowy interfejs użytkownika. Pakiet umożliwia korzystanie z danych zawartych w zewnętrznych plikach tekstowych (ASCII).

Wersję studencką pakietu można pobrać ze strony:
https://ampl.com/products/ampl/ampl-for-students/#Demo

Przygotowanie programu AMPL do pracy

Pierwszym krokiem przy korzystaniu z pakietu jest uruchomienie programu sw, który otwiera okno tekstowe (w pliku README.SW jest pełny opis poleceń programu SW).

W oknie poleceń sw należy wpisać polecenie ampl i nacisnąć Enter. Uruchomienie systemu AMPL potwierdzane jest zmianą symbolu w linii poleceń na ampl:. Od tej chwili polecenia są interpretowane przez program AMPL. Polecenia dla AMPL należy zawsze kończyć średnikiem.

Pracę z AMPL kończy się poleceniem quit; lub end; następuje wówczas powrót do okna tekstowego, z którego AMPL został wywołany. Pracę w oknie tekstowym kończy się przez zamknięcie okna, a w przypadku programu SW także poleceniem CTRL+Z.

Poniżej zostały opisane podstawowe komendy języka AMPL, definiowanie modelu zadania oraz definiowanie parametrów modelu.

Podstawowe komendy programu AMPL

Tabela 1 przedstawia podstawowe komendy jeęzyka AMPL. Na zielono zostały wyróżnione te komendy, które będą stosowane podczas wykonywania ćwiczenia laboratoryjnego.

Tabela 1. Podstawowe komendy języka AMPL

data	przejście do trybu pracy *data*; wstawienie pliku z danymi,
display	wypisanie wartości funkcji celu, zmiennych, ograniczeń modelu,
include	wstawienie pliku,
let	zmiana wartości danych,
model	przejście do trybu pracy *model*; wstawienie pliku z modelem,
objective	wybranie zmiennych do funkcji celu,
option	ustawienie lub wypisanie opcji,
quit	wyjście z AMPL,
reset	kasowanie modelu/danych (umożliwia wprowadzenie następnego),
solve	rozwiązanie zadania,
write	zapisanie problemu do plików na dysk.

Definiowanie modelu zadania

Podczas definiowania modelu zadania należy określić nazwy parametrów, nazwy i typ zmiennych, funkcję celu, ograniczenia w postaci wyrażeń, zbiory indeksów (opcjonalnie).

Ogólne zasady konstruowania modelu są następujące:
- każde wyrażenie musi być zakończone średnikiem: `;',
- komentarze muszą zaczynać się od znaku: #,
- wszystkie zmienne są domyślnie traktowane jako ciągłe,
- do konstrukcji wyrażeń są używane operatory: *, /, -, +
- kolejność wykonywania działań arytmetycznych jest zgodna z ogólnie przyjętymi zasadami,
- zmienne całkowite są dodatkowo określane w deklaracji jako: integer,

- zmienne binarne są dodatkowo określane w deklaracji jako: binary,
- symbole \infty oraz -\infty są określane w deklaracji jako: Infinity oraz –Infinity,
- indeksowanie danych:
  {*i* in *1..N,j* in *1..M*} - wszystkie pary (i,j) *i* od *1* do *N*,j od *1* do *M*, przy czym N,M zostały wcześniej zadeklarowane jako parametry,

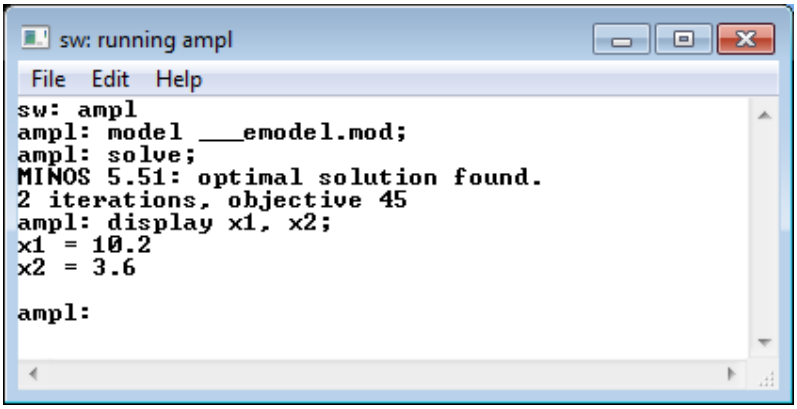
- zbiory i parametry są deklarowane odpowiednio poleceniami: set oraz param,
- zmienne są deklarowane poleceniem: var,
- funkcja celu jest deklarowana poleceniem: minimize lub maximize,
- ograniczenia są deklarowane poleceniem: subject to.

Najprostszy sposób zapisu i rozwiązania modelu matematycznego w języku AMPL.

W tabeli 2 pokazono sformułowanie matematyczne pewnego zagadnienia w postaci zadania programowania liniowego oraz zapis tego zadania jako modelu w języku AMPL. Model AMPL można zapisać w pliku tekstowym o rozszerzeniu mod (w przykładzie plik modelu ma nazwę: ___emodel.mod) i rozwiązać pisząc sekwencję poleceń tak, jak pokazuje rysunek 1.

Tabela 2. Przykładowy sposób zapisu ZPL w języku AMPL

Dane jest następujące ZPL:	#Zapis w języku AMPL
$\max 3x_1 + 4x_2$ przy ograniczeniach: $2x_1 + 3x_2 \leq 36$ $2x_1 + 6x_2 \leq 42$ $2x_1 + x_2 \leq 24$ $x_1, x_2 \geq 0$	**var** x1 >= 0; **var** x2 >= 0; **maximize** funkcja_celu**:** 3*x1 + 4*x2; **subject to** ograniczenie1**:** 2*x1 + 3*x2 <= 36; **subject to** ograniczenie2**:** 2*x1 + 6*x2 <= 42; **subject to** ograniczenie3**:** 2*x1 + x2 <= 24;



Rysunek 1. Rozwiązanie modelu zapisanego w pliku ___emodel.mod

Wykonanie polecenia solve; spowoduje uruchomienie standardowego solvera MINOS.

MINOS 5.5: optimal solution found.
2 iterations, objective 45

Do dyspozycji mamy również solver CPLEX - aby zmienić solver wykonać instrukcję:

option solver cplex;

Solver MINOS nie obsługuje zmiennych całkowitoliczbowych; robi to solver CPLEX.

Do obejrzenia wyników służy polecenie display.

display funkcja_celu, x1, x2;

Wyniki można zapisać do pliku dodając symbol > i nazwę pliku, np.:

display funkcja_celu, x1, x2 > plik_rozwiazania.out;

Rozszerzony sposób zapisu modelu i danych w języku AMPL.

Procesor języka AMPL rozróżnia pliki ze względu na rozszerzenia.

- plik **.mod* zawiera zapis modelu - deklaracje zmiennych, parametrów, funkcję celu oraz ograniczenia modelu;
- plik **.dat* zawiera wartości danych modelu;
- plik **.run* zawiera ciągi poleceń (plik batchowy);

Tabela 3. Zapis modelu i danych w osobnych plikach.

#Model – *. mod	#Dane – *. dat	#Plik poleceń - *.run

param N;
param M;
param c {1..N};
param b {1..M};
param a {1..M, 1..N};
 	
var x {j in 1..N} >= 0, integer;
maximize z: sum {j in 1..N} c[j]*x[j];
subject to ogr1 {i in 1..M}: sum{j in 1..N} a[i,j]*x[j] <= b[i];
	param N := 2; # lb. zmiennych
	param M := 3; # lb. ograniczeń
	param c :=
	1 3
	2 4 ; #wektor współ .funkcji celu
	param a :
	1 2 :=
	1 2 3
	2 2 6
	3 2 1 ; #macierz współ. ograniczeń
	param b :=
	1 36
	2 42
	3 24 ; #wektor współ. prawych stron ogran.
		option solver cplex;
		model *.mod;
		data *.dat
		solve;
		display z;
		display x;

Uwaga: W modelu przedstawionym w tabeli 2 nałożony został warunek całkowitoliczbowości na zmienne x.