Laboratorium WDEC

Opis posługiwania się pakietem AMPL

Adam Krzemienowski, Grzegorz Płoszajski
Instytut Automatyki i Informatyki Stosowanej
Politechnika Warszawska

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).

Przygotowanie programu AMPL do pracy

Pakiet AMPL działa w środowisku Windows. Pierwszym krokiem przy korzystaniu z pakietu jest uruchomienie programu sw z katalogu AMPL, który otwiera okno tekstowe (z symbolem 'sw:' — Scrolling Window — w linii poleceń). W pliku README.SW jest pełny opis poleceń programu SW. Można jednak zamiast z SW korzystać z innego okna tekstowego, np. z wiersza poleceń.

W oknie poleceń 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`' (ze średnikiem); 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.

Wstępne sprawdzenie działania AMPL

Można sprawdzić, czy program działa, wpisując następującą sekwencję poleceń:

```
ampl
model models\diet.mod;
data models\diet.dat;
solve;
```

Powinien pojawić się napis:

```
MINOS 5.5: optimal solution found.
6 iterations, objective 88.2
```

W przypadku rozwiązywania zadań ze zmiennymi całkowitoliczbowymi zamiast solvera MINOS należy użyć solvera CPLEX. W tym celu należy wydać polecenie:

```
option solver cplex;
```

W odpowiedzi na wydane następnie polecenie '`solve`' powinien się teraz pojawić komunikat:

```
CPLEX 11.2.0: optimal solution; objective 88.2
0 simplex iterations (0 in phase I)
```

W podobny sposób można zmienić solwer na LPSOLVE lub z powrotem na MINOS.

Język AMPL

Język AMPL (A Mathematical Programming Language) jest algebraicznym językiem modelowania problemów programowania liniowego, nieliniowego lub całkowitoliczbowego. Poniżej zostały opisane podstawowe komendy języka AMPL, definiowanie modelu zadania oraz definiowanie parametrów modelu.

Podstawowe komendy programu AMPL

Do podstawowych komend należą:

`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`,
- równoważne są dwa sposoby indeksowania danych:

  {*i* in *A*,*j* in B} - wszystkie pary (i,j) i z A, j z B, przy czym A,B zostały wcześniej zadeklarowane jako zbiory,

  {*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`.

Sposób zapisu ZPL w języku AMPL

Dane jest następujące ZPL:

```
max 120 X1+115 X2
przy ograniczeniach:\\
12 X1 – 3 X2 ≤ 72
–X1 + 5 X2 ≤ 143
2 X1– 3 X2 ≤ 21
13 ≤ X1 ≤ 150
92 ≤ X2 ≤ 912
X1, X1 ≥ 0, całkowite
```

Jego model w języku AMPL wygląda następująco:

```
param N;
param M;
param c {1..N};
param l {1..N};
param u {1..N};
param b {1..M};
param a {1..M, 1..N};
var x {j in 1..N} >= l[j], <= u[j], integer;
maximize f_celu: 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];
subject to ogr2 {j in 1..N}: x[j]>=0;
```

W powyższym przykładzie przy pomocy deklaracji `param` zadeklarowane zostały pojedyncze parametry: *N* i *M*, wektory parametrów: *c,l,u* i *b*, indeksowane odpowiednio od 1 do *N* i od 1 do *M*, oraz macierz parametrów *a{M* x *N}* (mająca *M* wierszy i *N* kolumn).

Wektor zmiennych x indeksowany od 1 do *N* został zadeklarowany poleceniem `var`, przy czym narzucone zostały proste ograniczenia dolne i górne na zmienne *x* i warunek całkowitoliczbowości.

Suma po *j* od 1 do *N* iloczynów $c_j x_j$ została zapisana przy pomocy polecenia `sum` i wprowadzona do funkcji celu *f_celu* poleceniem `maximize`.

Ograniczenia modelu zostały zapisane przy pomocy poleceń `subject to`, przy czym *ogr1* to *M* nierówności (*i* zmienia się od 1 do *M*), z których każda stanowi sumę po *j* od 1 do *N* iloczynów $a_{ij} x_j$ niewiększą niż b_i (dla *i*-tego ograniczenia).

Ograniczenia *ogr2* zawierają warunek nieujemności zmiennych x_j dla *j* od 1 do *N*.

Uwaga, AMPL rozróżnia małe i duże litery.

Definiowanie parametrów modelu

Biorąc pod uwagę sposób zapisu modelu zadania przedstawiony w poprzednim punkcie, jego parametry można zdefiniować następująco:

```
param N := 2;               # liczba zmiennych
param M := 3;               # liczba ograniczeń
param c :=
           1 120
           2 115 ;          # wektor współczynników funkcji celu
param l :=
           1  13
           2  92 ;          # proste ograniczenia dolne na zmienne
param u :=
           1  150
           2  912 ;         # proste ograniczenia górne na zmienne
param a :
             1     2    :=
      1     12    -3
      2     -4     5
      3      2    -3 ;       # macierz współczynników ograniczeń
param b :=
           1  72
           2  143
           3  21 ;           # wektor współczynników prawych stron
```

Wektorom parametrów *c,l,u,b* przypisano wartości na odpowiednich pozycjach, podając kolejno: numer pozycji oraz wartość parametru na tej pozycji.

Macierzy parametrów *a* przypisano wartości podając po dwukropku numery kolumn macierzy, a następnie po znaku przypisania (:=) kolejno dla każdego wiersza: numer wiersza macierzy, wartości parametrów w kolejnych kolumnach.

Przykładowy sposób rozwiązania problemu

Procesor języka AMPL rozróżnia pliki ze względu na rozszerzenia.

Dla użytkownika najważniejsze są dwa z nich:

- 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);

wszystkie będące plikami tekstowymi.

Należy zatem zapisać model w pliku tekstowym *nazwa1*.mod, a parametry w pliku *nazwa2*.dat, uruchomić program AMPL i wczytać pliki z modelem i danymi poleceniami:

```
model nazwa1.mod;
data nazwa2.dat;
```

Jeżeli nie pojawią się komunikaty błędów, można przystąpić do rozwiązania modelu. Polecenie:

```
solve;
```

spowoduje uruchomienie standardowego solwera MINOS.

```
MINOS 5.5: ignoring integrality of 2 variables
MINOS 5.5: optimal solution found.
3 iterations, objective 20885
```

Solwer ten nie obsługuje zmiennych całkowitoliczbowych. Gdyby zatem rozwiązanie nie było całkowitoliczbowe, należałoby zmienić solwer na CPLEX poleceniem:

```
option solver cplex;
```

i ponownie rozwiązać problem poleceniem `solve`.

Do obejrzenia wyników służy polecenie `display.` Należy w nim wymienić nazwy zmiennych, funkcji, parametrów, np. `x, f_celu`.

```
display f_celu, x, c;
```

Wyniki można zapisać do pliku dodając przed średnikiem symbol '`>`' i nazwę pliku, np.:

```
display f_celu, x, c > nazwa3.out;
```

Zbiory

Uwaga, w przypadku zmiennych i parametrów wektorowych poszczególne składowe mogą być identyfikowane nazwami, a nie numerami. Zamiast x[1], x[2] na oznaczenie ilości produkowanych produktów rolnych można operować nazwami kojarzącymi się z poszczególnymi uprawami: ziemniaki, buraki. W modelu nie musi się operować rozmiarem wektorów, lecz bezpośrednio zbiorami i indeksami. Na przykład

Model:

```
set PROD;
param profit {PROD};
param market {PROD};
param rate {PROD};
param dostep;
var Uprawa {p in PROD} >=0, <=market[p];
maximize total_profit: sum {p in PROD} profit[p]*Uprawa[p];
subject to ogr: sum {p in PROD} (1/rate[p])*Uprawa[p]<=dostep;
```

Dane:

```
set PROD := ziemniaki  buraki;
param:            rate        profit      market  :=
ziemniaki         200         25          6000
buraki            140         30          4000 ;
param dostep := 40;
```

Jeżeli poprzednio był rozwiązywany inny model, należy najpierw wydać polecenie reset. Po wprowadzeniu modelu i danych zapisanych jak wyżej i rozwiązaniu problemu, w przypadku solvera MINOS potwierdzonym komunikatem:

```
MINOS 5.5: optimal solution found.
2 iterations, objective 192000
```

można odczytać rozwiązanie poleceniem:

```
display Uprawa;
```

W odpowiedzi otrzyma się komunikat:

```
Uprawa [*] :=
   buraki  1400
ziemniaki  6000
;
```