

优化软件与应用

主讲人： 雒兴刚
东北大学系统工程研究所
Email: luoxinggang@ise.neu.edu.cn
Tel: 83682292



信息科学与工程学院
COLLEGE OF INFORMATION SCIENCE AND ENGINEERING

第八章 ILOG OPL进阶功能

ILOG 数据库连接

1、与ODBC数据源的连接

OPL支持通过**ODBC**数据源读取数据库中的数据，下面是数据库连接的语句，都以大写的**DB**开头：

```
Dbconnection db("odbc","DRIVER={Microsoft Access Driver  
(* .mdb)};DBQ=..\examples\opl\oilDB.mdb//");
```

功能：打开数据库，建立数据库连接；

Db---数据库连接的名字；

“**odbc**”---指明以**ODBC**方式连接；**OPL**也同时支持**Oracle**等**RDB**直连；

“**Driver=...**”---连接字符串；不同数据库对应的连接字符串不同，有的需要提供用户名称和密码，具体请参阅**ODBC**文档；



第八章 ILOG OPL进阶功能

ILOG 数据库连接

Mod文件:

```
{string} Gasolines = ...;
```

Db必须通过前面的
DBconnection初始化完毕

Dat文件:

```
DBconnection db("odbc","DRIVER={Microsoft Access Driver  
(* .mdb)};DBQ=..\examples\opl\oilDB.mdb//");
```

```
Gasolines from DBread(db,"SELECT name FROM GasData");
```

GasData : Table					
	name	demand	price	octane	lead
▶	Super	3000	70	10	1
	Regular	2000	60	8	2
	Diesel	1000	50	6	1
*					
Record: 1 of 3					

第八章 ILOG OPL进阶功能

ILOG 数据库连接

Mod文件:

tuple GasType

{

string name;

float demand;

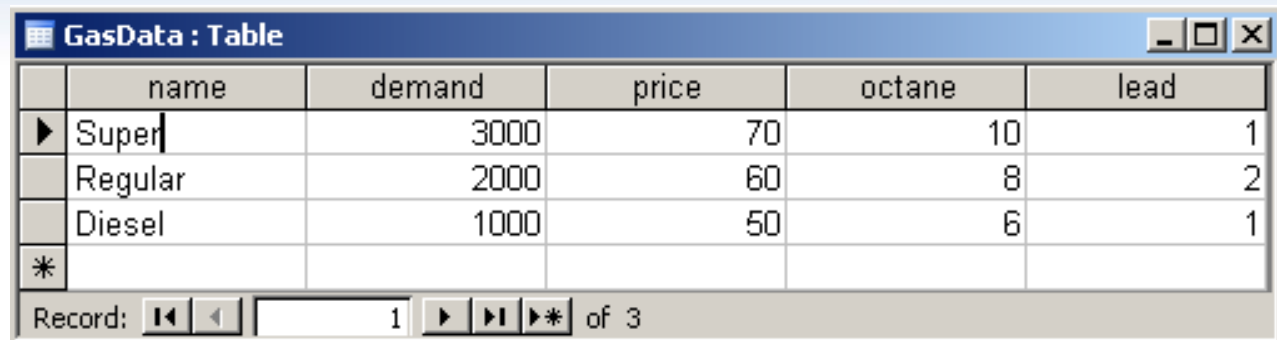
float price;

float octane;

float lead;

}

{GasType} gasData = ...;



	name	demand	price	octane	lead
▶	Super	3000	70	10	1
	Regular	2000	60	8	2
	Diesel	1000	50	6	1
*					

Record: 1 of 3

Dat文件:

DBconnection db("odbc","DRIVER={Microsoft Access Driver (*.mdb)};DBQ=..\examples\opl\oilDB.mdb//");

gasData **from** **DBread**(db,"SELECT * FROM GasData");

注意：字段名称要和**tuple**定义成员的名称相同！



信息科学与工程学院
COLLEGE OF INFORMATION SCIENCE AND ENGINEERING

第八章 ILOG OPL进阶功能

利用**DB**语句删除表: **ILOG 数据库连接**

DBexecute(db,"drop table Result");

新建一个表:

DBexecute(db,"create table Result(oil varchar(10), gas varchar(10), blend real, a real)");

输出运行结果到数据库:



第八章 ILOG OPL进阶功能

ILOG 数据库连接

例如，mod文件为

```
tuple Result {  
    string oil;  
    string gas;  
    float blend;  
    float a;  
}  
{string} Oils = ...; {string} Gasolines = ...;  
{Result} result = { <o,g,1,0> | o in Oils, g in Gasolines };  
execute{  
    writeln("Name of Oils: ",Oils);  
    writeln("Name of Gasolines: ",Gasolines);  
}
```

Dat文件为

```
DBconnection db("odbc","DRIVER={Microsoft Access Driver  
(* .mdb)};DBQ=c:\\oilDB.mdb//");  
Gasolines from DBread(db,"SELECT name FROM GasData");  
Oils from DBread(db,"SELECT name FROM OilData");  
result to DBupdate(db,"INSERT INTO Result(oil,gas,blend,a) VALUES(?,?,?,?)");
```



第八章 ILOG OPL进阶功能

ILOG 数据库连接

2、与Excel的连接

OPL通过SheetConnection连接Excel表:

SheetConnection sheet("e:\\oilSheet.xls");

功能：打开数据库，建立数据库连接；

“oilSheet.xls”---文件名称，路径是OPL运行路径；也可以给出全路径；
sheet---连接的名称；



第八章 ILOG OPL进阶功能

ILOG 数据库连接

读取Excel里的数据

Mod文件:

{string} Gasolines = ...;

Dat文件:

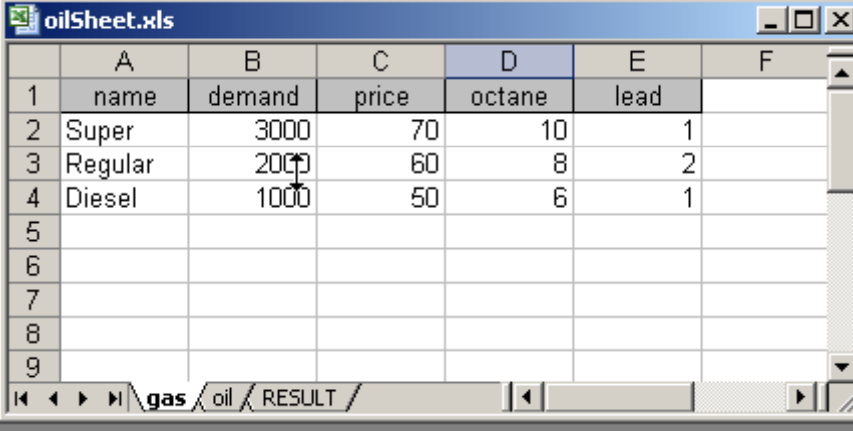
SheetConnection sheet("e:\\oilSheet.xls");

Gasolines from SheetRead(sheet,"gas!A2:A4");

运行后，查看Model Browser中的Gasolines

sheet必须通过前面的
DBconnection初始化完毕

注意 “gas!A2:A4”中，gas表示Excel的work sheet的名字，A2:A4表示取这个范围内的单元格。



	A	B	C	D	E	F
1	name	demand	price	octane	lead	
2	Super	3000	70	10	1	
3	Regular	2000	60	8	2	
4	Diesel	1000	50	6	1	
5						
6						
7						
8						
9						

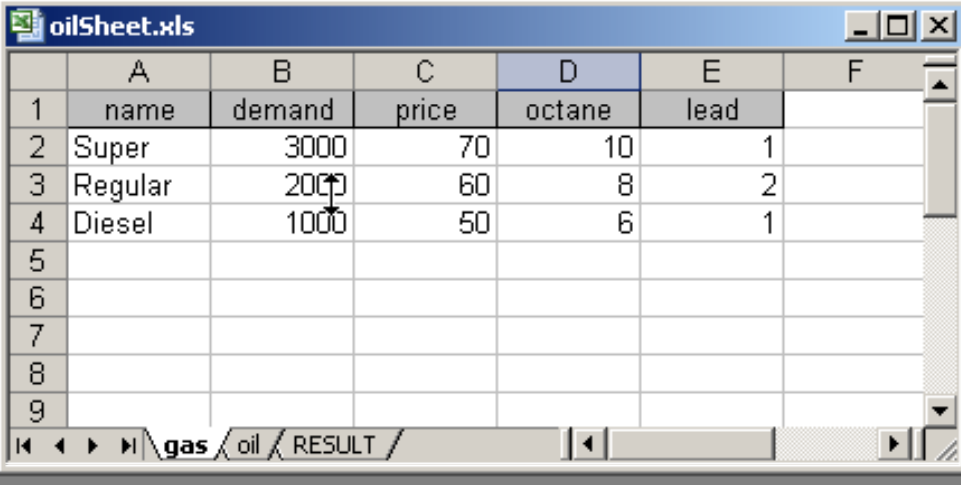


第八章 ILOG OPL进阶功能

ILOG 数据库连接

Mod文件:

```
tuple GasType
{
    float demand;
    float price;
    float octane;
    float lead;
}
{GasType} gasData = ...;
{int}blend={1,2};
```



The screenshot shows an Excel spreadsheet titled 'oilSheet.xls'. The data is organized in a table with columns A through F. Column A contains gas names, B contains demand, C contains price, D contains octane, and E contains lead. The first four rows contain data for Super, Regular, and Diesel gas, with the fourth row being empty. The bottom of the spreadsheet shows a navigation bar with tabs for 'gas', 'oil', and 'RESULT'.

	A	B	C	D	E	F
1	name	demand	price	octane	lead	
2	Super	3000	70	10	1	
3	Regular	2000	60	8	2	
4	Diesel	1000	50	6	1	
5						
6						
7						
8						
9						

Dat文件:

```
SheetConnection sheet("e:\\oilSheet.xls");
gasData from SheetRead(sheet,"gas!B2:E4");
```

类似的，可以用SheetWrite将数据写到Excel:

```
blend to SheetWrite(sheet,"RESULT!B2:B3");
```



信息科学与工程学院
COLLEGE OF INFORMATION SCIENCE AND ENGINEERING

第八章 ILOG OPL进阶功能

ILOG Script 介绍

OPL语言主要着眼于模型的建模、表达约束和目标等。但是，优化应用很多时候需要更强的功能去控制算法求解的流程和更强的处理数据的方法。

作为一种非建模语言，**ILOG** 脚本提供了这方面的功能。

主要的功能包括：

- ✓ 数据前处理；
- ✓ 数据后处理；
- ✓ 流程控制；
- ✓ ...



第八章 ILOG OPL进阶功能

ILOG Script介绍

ILOG 脚本和**OPL**结合主要有两种方式，一种是嵌入**Execute**语句，一种是用**main**函数。二者都在**mod**文件中。前者主要是用于设置参数、数据的前处理（如数组初始化）和后处理，后者可以提供很强的综合能力，例如参数设置、数据处理、控制程序运行流程、设置改变模型参数重新运行模型等等。

例如，可以用下面的**Execute**语句设置**Options**里的参数（仅对当前运行的模型有效）：

```
execute CPX_PARAM {  
    cplex.preind = 0;  
    cplex.simdisplay = 2;  
}
```

注意**CPX_PARAM**不是关键字，是**execute**块的名字，同一个文件里的**execute**块不要重名就可以了。

第八章 ILOG OPL进阶功能

ILOG Script 介绍

下面的语句输出一行Hello World提示信息:

```
execute HELLO {  
  writeln("Hello World.");  
}
```

运行后在Output窗口的Console标签页会看到 “Hello world”字样。

另外， **execute** 语句在模型中的位置可以在约束和目标之前，也可以在后。如果在后，就在模型求解之前运行；如果之后，则在模型求解后运行；

可以用于数组的初始化。例如：

```
int nNum=10;  
range rSample=1..nNum;  
float fMyArray[rSample];  
execute My_Init{  
  for(var i=1; i<=nNum; i++)  
  {    fMyArray[i]=i*i;    writeln("Array",i,"=",fMyArray[i]);  
  }  
}
```



第八章 ILOG OPL进阶功能

ILOG Script 介绍

结果输出以下信息:

Array1=1

Array2=4

Array3=9

Array4=16

Array5=25

Array6=36

Array7=49

Array8=64

Array9=81

Array10=100

这里用到了ILOG脚本的for语句，语法和C语言的用法类似。



第八章 ILOG OPL进阶功能

ILOG Script 介绍

另外一种方式就是使用**main**函数。如果**mod**文件中含有**main**函数，**IDE**就会：

1. 执行约束和目标前的**execute**语句；
2. 不自动执行模型求解，也不会执行约束和目标后的**execute**语句；
3. 执行**main**函数，然后结束；

例如，可以在一个打开的**mod**文件中输入一个**main**函数：

```
main{  
    thisOplModel.generate();  
}
```

这里的**thisOplModel**是脚本的一个关键字，表示当前**mod**文件里的模型和附带的**dat**文件数据（如果有**dat**文件）。



第八章 ILOG OPL进阶功能

ILOG Script 介绍

可以尝试下面的main函数:

```
main{  
    thisOplModel.generate();  
    with (thisOplModel) { writeln("Hello World.");}  
}
```

和**execute HELLO** {writeln("Hello World.");}功能相同。

另一个重要的关键字是**cplex**，表示当前模型对应的算法实例。参见下例：

```
main{  
    thisOplModel.generate();  
    with (thisOplModel) { writeln("Hello World.");}  
    cplex.solve() ;  
    curr = cplex.getObjValue();  
    writeln("Optimal=: ",curr);  
}
```



第八章 ILOG OPL进阶功能

ILOG Script 介绍

也可以在main中，根据已有的mod和dat文件，重新建立新的模型，例如：

```
main{
  var mySrc=new IloOplModelSource('e:\\production.mod');
  var myDef = new IloOplModelDefinition(mySrc);
  var myCplex = new IloCplex();
  var myOpl = new IloOplModel(myDef,myCplex);
  var myData = new IloOplDataSource('e:\\production.dat');
  myOpl.addDataSource(myData);
  myOpl.generate();

  myCplex.solve();
  myOpl.postProcess();

  var curr = myCplex.getObjValue();
  writeln('Optimal=: ',curr);
}
```



第八章 ILOG OPL进阶功能

ILOG Script 介绍

脚本也支持函数的定义和调用，语法和C语言类似，但不支持递归和嵌套。

例如，下面的定义了一个打印的函数，并在后面调用：

```
function myPrint (myPara){  
    for(v in myOpl.Products){  
        writeln("inside ['v,']="myPara.inside[v]);  
        writeln("outside ['v,']="myPara.outside[v]);  
    }  
}  
myPrint(myOpl);
```

同时，可以设置断点，并用step into、step outside等命令进行调试。



第八章 ILOG OPL进阶功能

ILOG Script 介绍

下面是一个循环调用求解的例子。每次修改面粉的capacity，重新求解模型。

```
main{
    var mySrc=new IloOplModelSource("e:\\production.mod"); Starting Script execution...
    var myDef = new IloOplModelDefinition(mySrc);           Optimal=: 372
    var myCplex = new IloCplex();                           Optimal=: 368
    var myOpl = new IloOplModel(myDef,myCplex);              Optimal=: 364
    var myData = new IloOplDataSource("e:\\production.dat"); Optimal=: 360
    myOpl.addDataSource(myData);                             Optimal=: 356.67
    myOpl.generate();                                       Optimal=: 355
    for(var v=1; v<=10;v++){                               Optimal=: 355
        myCplex.solve();                                    Optimal=: 355
        var curr = myCplex.getObjValue();                   Optimal=: 355
        writeln("Optimal=: ",curr);                          Optimal=: 355
        var myBidingData=myOpl.dataElements;                Optimal=: 355
        myBidingData.capacity["flour"]+=10;                 Optimal=: 355
        myOpl=new IloOplModel(myDef,myCplex);               Optimal=: 355
        myOpl.addDataSource(myBidingData);                  Finished Script execution
        myOpl.generate();
    }
}
```

本例未加出错处理！



第八章 ILOG OPL进阶功能

ILOG Script 介绍

为了在迭代过程中使用上次求解的信息，可以使用**CplexBasis**，程序修改如下：

```
main{  
var myBasis=new IloOplCplexBasis();  
var mySrc=new IloOplModelSource("production.mod");  
var myDef = new IloOplModelDefinition(mySrc);  
var myCplex = new IloCplex();  
var myOpl = new IloOplModel(myDef,myCplex);  
var myData = new IloOplDataSource("production.dat");  
myOpl.addDataSource(myData);  
myOpl.generate();  
for(var v=1; v<=10;v++){  
    myCplex.solve();  
    curr = myCplex.getObjValue();  
    writeln("Optimal=: ",curr);  
    myBasis.getBasis(myCplex);  
    var myBidingData=myOpl.dataElements;  
    myBidingData.capacity["flour"]+=10;  
    myOpl=new IloOplModel(myDef,myCplex);  
    myOpl.addDataSource(myBidingData);  
    myOpl.generate();  
    myBasis.setBasis(myCplex);  
}  
}
```



第八章 ILOG OPL进阶功能

ILOG Script 介绍

使用脚本可以输出运行结果到外部文件。例如用下面的语句创建一个文件：

```
var ofile = new IloOplOutputFile("mulprod_main.txt");
```

然后用**writeln**语句将变量的值写到文件中：

```
ofile.writeln("Objective with capFlour = ", capFlour, " is ", curr);
```

关闭文件使用语句：

```
ofile.close();
```



第八章 ILOG OPL进阶功能

OPL 和高级语言的接口

ILOG 提供和**C++**、**.net**、**java**等高级语言的接口，可以由高级语言调用**OPL**的各种功能实现。其语法与函数和**ILOG**脚本相似。

如果想利用**VC++6**以下版本连接**OPL**，需要单独制作和封装**dll**。直接包含**OPL**头文件会导致新老版本**stream**文件的兼容性冲突。

用**.net**连接**OPL**很简单，只需要将 **ILOG/OPL42/Lib**下的**oplall.dll**文件**reference**到工程中，然后在代码文件中加上：

```
using ILOG.Concert;  
using ILOG.CPLEX;  
using ILOG.OPL;
```



第八章 ILOG OPL进阶功能

OPL 和高级语言的接口

例如，下面用.net里的C#实现读取production.mod 和production.dat 建立模型、进行求解并输出结果：

```
OplFactory myFct = new OplFactory();
OplErrorHandler myErrHandler = myFct.CreateOplErrorHandler(Console.Out);

OplCplexBasis myBasis=myFct.CreateOplCplexBasis();
OplModelSource mySrc = myFct.CreateOplModelSource("e:\\production.mod",false);
OplModelDefinition myDef = myFct.CreateOplModelDefinition(mySrc, myErrHandler);
Cplex myCplex = myFct.CreateCplex();
OplModel myOpl = myFct.CreateOplModel(myDef,myCplex);
OplDataSource myData = myFct.CreateOplDataSource("e:\\production.dat");
myOpl.AddDataSource(myData);
myOpl.Generate();
for (int i = 1; i <= 10; i++)
{
    myCplex.Solve();
    Console.Out.WriteLine("OBJECTIVE=: " + myOpl.Cplex.ObjValue);
    myOpl.PostProcess();
    myOpl.PrintSolution(Console.Out);
    myBasis.GetBasis(myCplex);
    OplDataElements myBidingData = myOpl.MakeDataElements();
    double flr_cap= myBidingData.GetElement("capacity").AsNumMap().Get("flour");
    myBidingData.GetElement("capacity").AsNumMap().Set("flour", flr_cap+10.0); ;

    myOpl = myFct.CreateOplModel(myDef, myCplex);
    myOpl.AddDataSource(myBidingData);
    myOpl.Generate();
    myBasis.SetBasis(myCplex);
}
myFct.End();
```

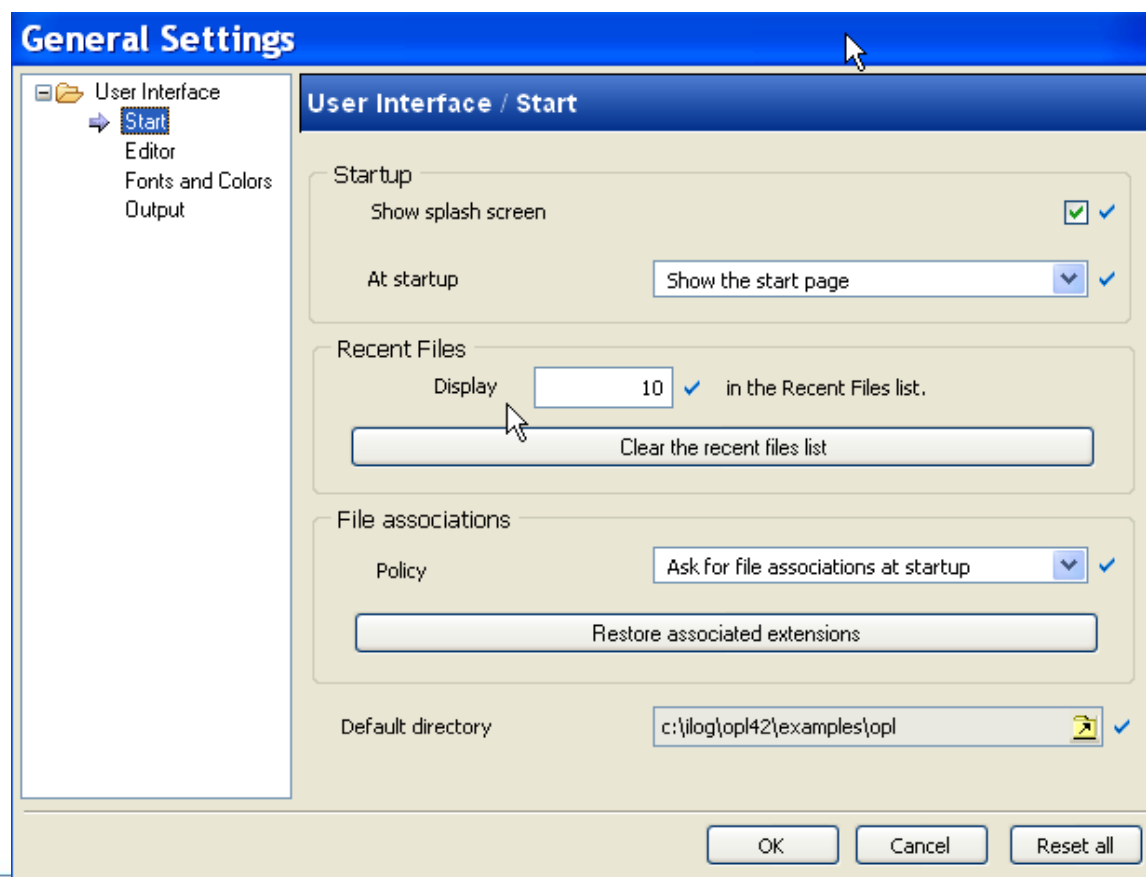


第八章 ILOG OPL进阶功能

ILOG 选项卡

1、界面和语言选项（Options>General Settings）

设置图形显示界面、编辑区特性、OPL的显示等。



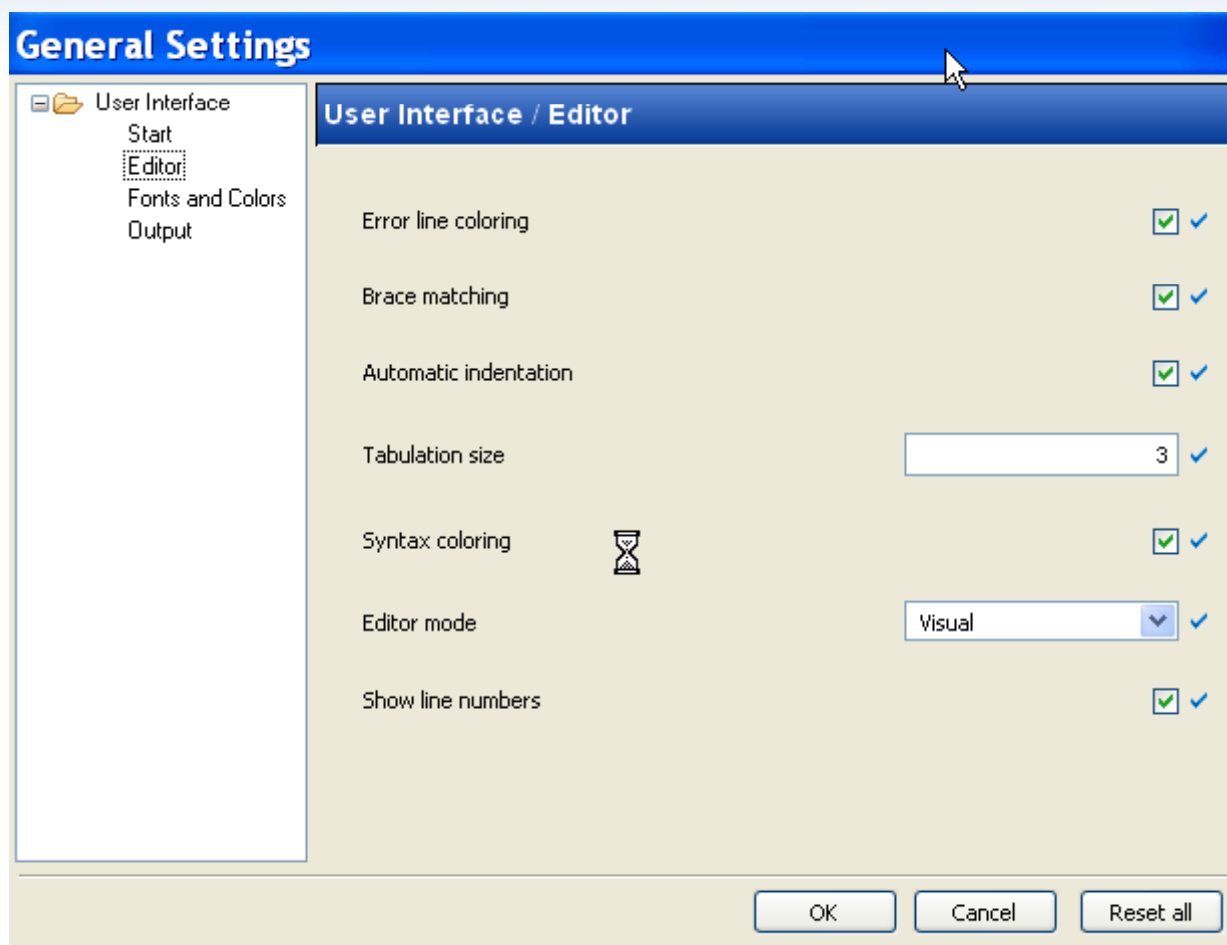
- ✓ 启动;
- ✓ 最近文件;
- ✓ 文件关联;
- ✓ 缺省目录;



第八章 ILOG OPL进阶功能

ILOG 选项卡

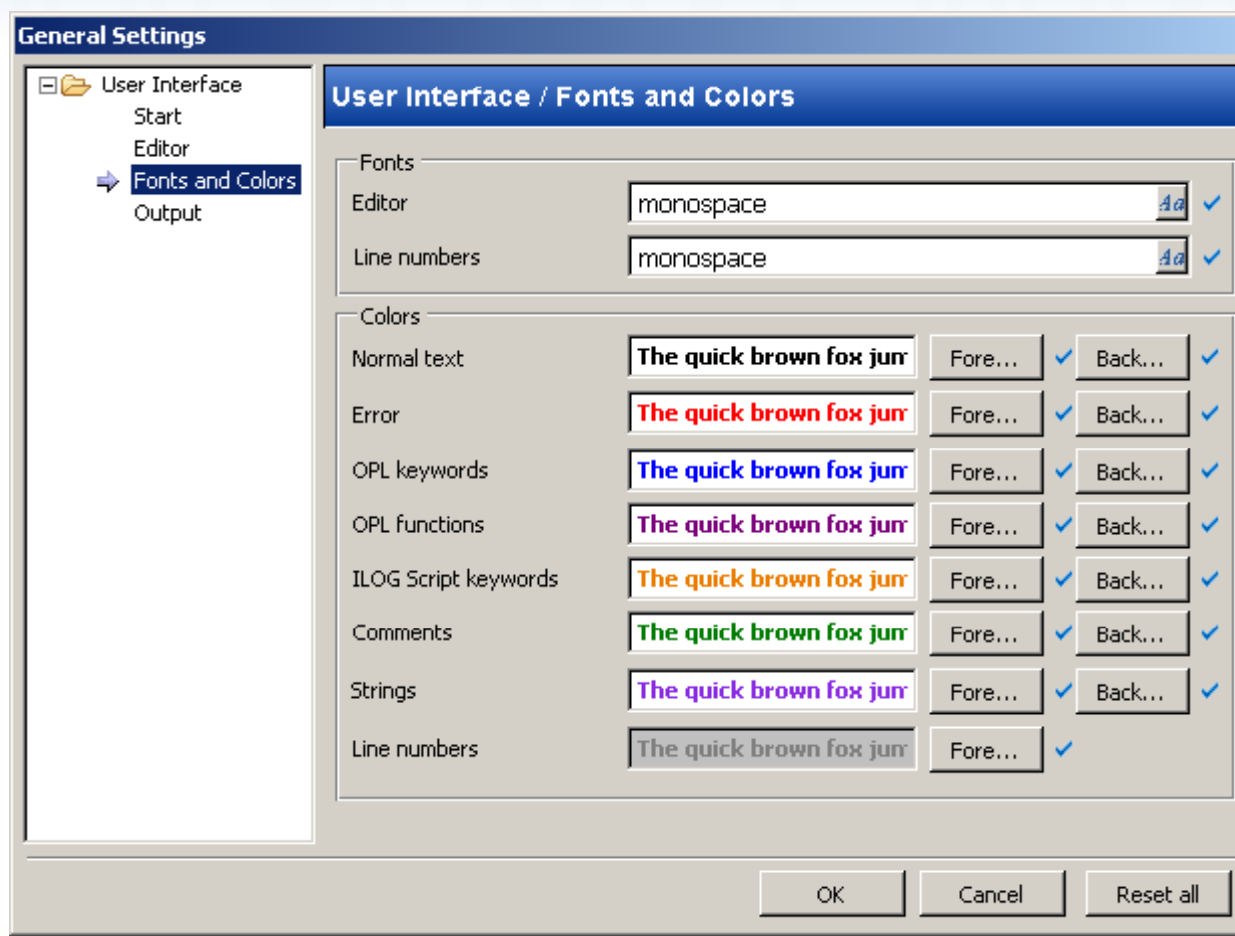
1、界面和语言选项（Options>General Settings）



第八章 ILOG OPL进阶功能

ILOG 选项卡

1、界面和语言选项（Options>General Settings）



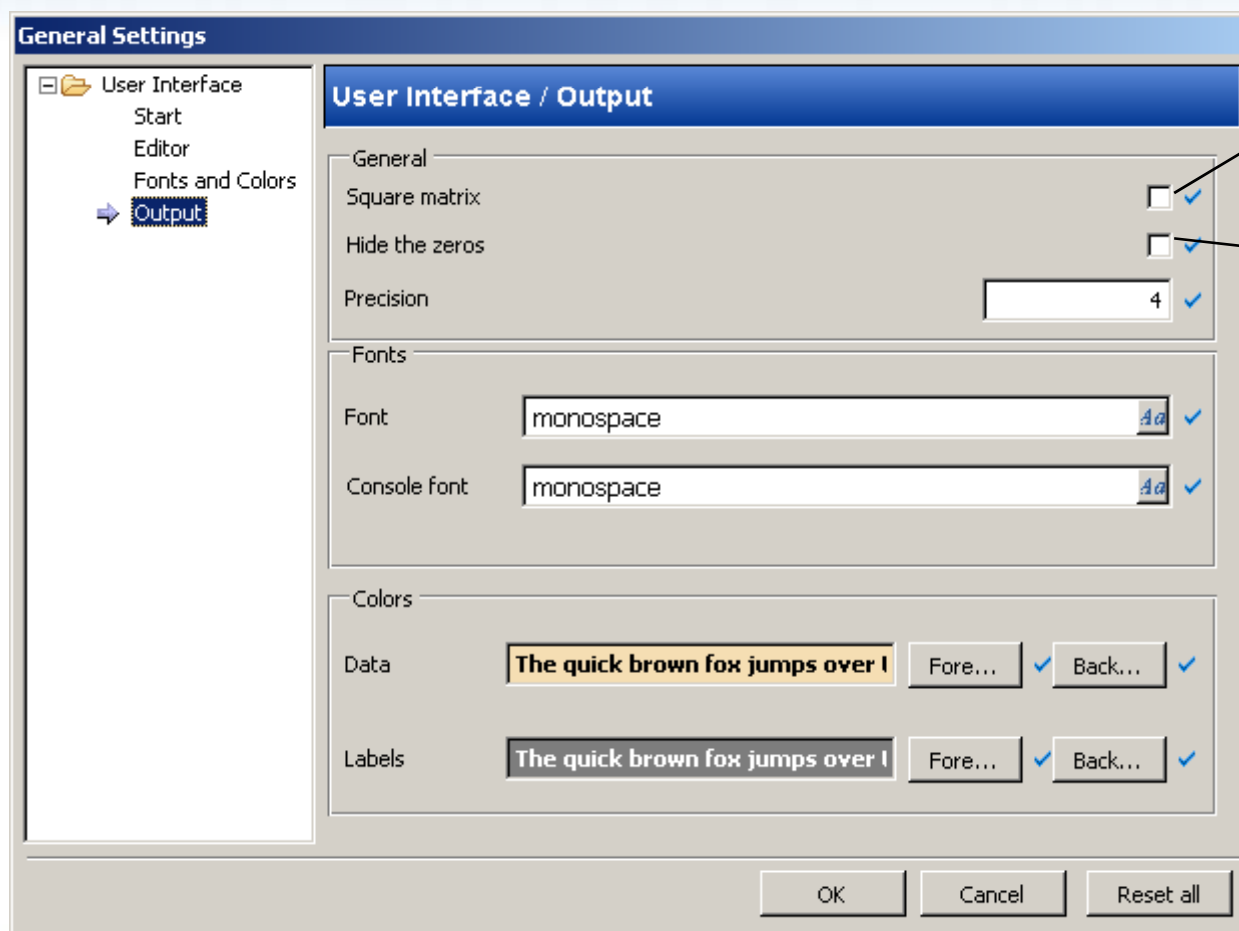
显示的都是用于预览的sample font！



第八章 ILOG OPL进阶功能

ILOG 选项卡

1、界面和语言选项（Options>General Settings）



Displayed matrix 是否用正方形显示;

Displayed matrix 中的零是否显示;

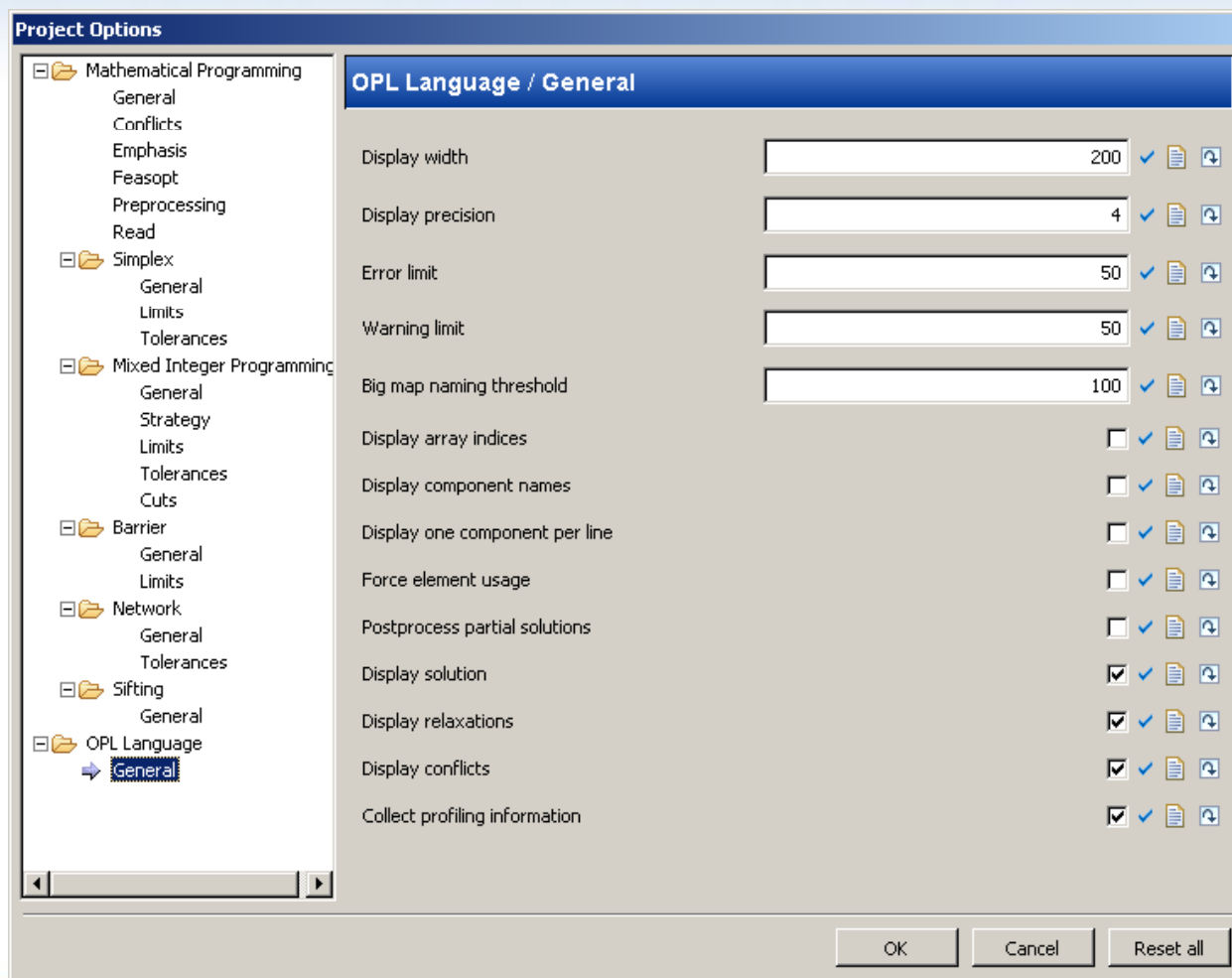


信息科学与工程学院
COLLEGE OF INFORMATION SCIENCE AND ENGINEERING

第八章 ILOG OPL进阶功能

ILOG 选项卡

1、界面和语言选项（Options>Project Options 最下边）



第八章 ILOG OPL进阶功能

ILOG 选项卡

Display width: 每行多少字符;

Display precision : 显示小数点后的位数;

Error limit : 显示最多的错误数量;

Big map naming threshold : 超过此值, 输出数组按照“大图”方式(即一种简略的显示方式);

Display array indices : 是否显示数组下标;

Display component names : tuple里的成员名称是否显示;

Display one component per line : 成员是否占一行;

Force element usage : 如果选中, 则没用的变量会自动警示;

Postprocess partial solutions: partial solutions也参与Postprocess;

Display solution : 如果不选, 则只显示目标值;

Display relaxations: 如果无解, 显示CPLEX建议的松弛方案;

Display conflicts: 如果无解, 显示检测到的冲突;

Collect profiling information: 一定要打开, 否则看不到Profiler标签里的信息。

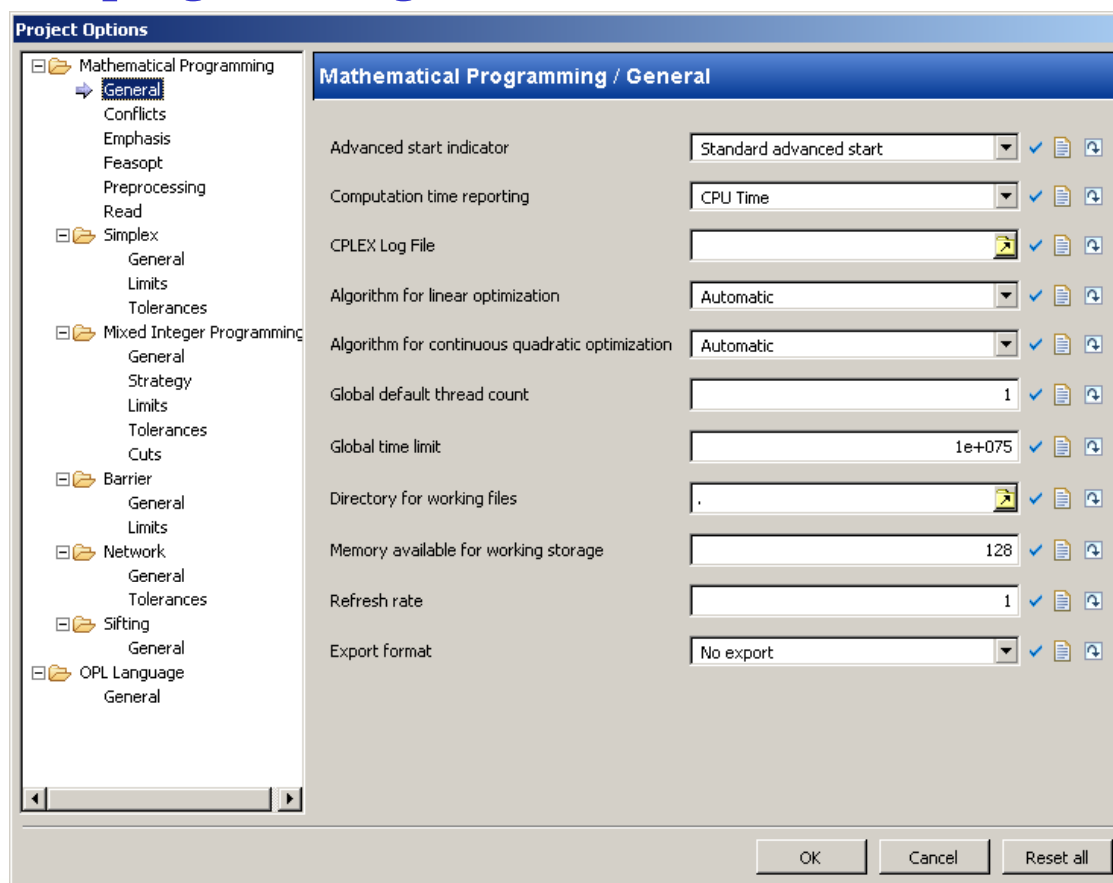


第八章 ILOG OPL进阶功能

ILOG 选项卡

2、数学规划选项（Options>Project Options）

Mathematical programming->General



第八章 ILOG OPL进阶功能

ILOG 选项卡

Advanced start indicator:

0 Do not use advanced start information

1 CPLEX will use an advanced basis supplied by the user

2 CPLEX will crush an advanced basis or starting vector supplied by the user

后面两个用于给定一个部分搜索的结果，在此基础上继续优化。前者使用一个上次计算（或给定的）基，后者使用一个启动向量。

Computation time reporting :

1 CPU time

2 Wall clock time (total physical time elapsed)

第二个更准确。



信息科学与工程学院
COLLEGE OF INFORMATION SCIENCE AND ENGINEERING

第八章 ILOG OPL进阶功能

ILOG 选项卡

CPLEX Log File: 设定Log文件的名称和路径;

Algorithm for linear optimization :

0 [CPX_ALG_AUTOMATIC] Automatic: let CPLEX choose

1 [CPX_ALG_PRIMAL] Primal Simplex (基本单纯型)

2 [CPX_ALG_DUAL] Dual Simplex (对偶单纯型)

3 [CPX_ALG_NET] Network Simplex (网络单纯型)

4 [CPX_ALG_BARRIER] Barrier (内点法)

5 [CPX_ALG_SIFTING] Sifting (过滤法)

6 [CPX_ALG_CONCURRENT] Concurrent (Dual, Barrier, and Primal)

设定CPLEX线性算法。



第八章 ILOG OPL进阶功能

ILOG 选项卡

Algorithm for continuous quadratic optimization :

0 [CPX_ALG_AUTOMATIC] Automatic: let CPLEX choose

1 [CPX_ALG_PRIMAL] Primal Simplex

2 [CPX_ALG_DUAL] Dual Simplex

3 [CPX_ALG_NET] Network Simplex

4 [CPX_ALG_BARRIER] Barrier

5 [CPX_ALG_SIFTING] Sifting

6 [CPX_ALG_CONCURRENT] Concurrent (Dual, Barrier, and Primal)

连续二次优化的算法选择。



信息科学与工程学院
COLLEGE OF INFORMATION SCIENCE AND ENGINEERING

第八章 ILOG OPL进阶功能

ILOG 选项卡

Global default thread count : CPLEX并行计算线程的数量;
Global time limit : 优化器的最大时间(秒), 超过则跳出优化;
Directory for working files : 存放临时文件的工作路径;
Memory available for working storage : CPLEX使用的工作内存上限(M);
Refresh rate: 显示刷新率(秒);
Export Format: 输出数据文件的格式;

Mathematical programming->Conflicts

Conflict information display :

0 no display

1 summary display

2 detailed display

决定CPLEX如何报告冲突。



信息科学与工程学院
COLLEGE OF INFORMATION SCIENCE AND ENGINEERING

第八章 ILOG OPL进阶功能

ILOG 选项卡

Global default thread count : CPLEX并行计算线程的数量;
Global time limit : 优化器的最大时间(秒), 超过则跳出优化;
Directory for working files : 存放临时文件的工作路径;
Memory available for working storage : CPLEX使用的工作内存上限(M);
Refresh rate: 显示刷新率(秒);
Export Format: 输出数据文件的格式;

Mathematical programming->Emphasis

Memory emphasis: 强调节省内存;
Numerical emphasis: 强调计算精度;
MIP emphasis indicator : 不同的强调策略

- 0 [CPX_MIPEMPHASIS_BALANCED] Balance optimality and feasibility
- 1 [CPX_MIPEMPHASIS_FEASIBILITY] Emphasize feasibility over optimality
- 2 [CPX_MIPEMPHASIS_OPTIMALITY] Emphasize optimality over feasibility
- 3 [CPX_MIPEMPHASIS_BESTBOUND] Emphasize moving best bound
- 4 [CPX_MIPEMPHASIS_HIDDENFEAS] Emphasize finding hidden feasible solutions



第八章 ILOG OPL进阶功能

ILOG 选项卡

Mathematical programming->Feasopt

Mode of FeasOpt : 对于一个不可行模型, 找最优松弛方案的策略

MinSum CPX_FEASOPT_MIN_SUM

Minimize the sum of all required relaxations in first phase only

1 OptSum CPX_FEASOPT_OPT_SUM

Minimize the sum of all required relaxations in first phase and execute second phase to find optimum among minimal relaxations

2 MinInf CPX_FEASOPT_MIN_INF

Minimize the number of constraints and bounds requiring relaxation in first phase only

3 OptInf CPX_FEASOPT_OPT_INF

Minimize the number of constraints and bounds requiring relaxation in first phase and execute second phase to find optimum among minimal relaxations

4 MinQuad CPX_FEASOPT_MIN_QUAD

Minimize the sum of squares of required relaxations in first phase only

5 OptQuad CPX_FEASOPT_OPT_QUAD

Minimize the sum of squares of required relaxations in first phase and execute second phase to find optimum among minimal relaxations



清华大学
COLLEGE OF INFORMATION SCIENCE AND ENGINEERING

第八章 ILOG OPL进阶功能

ILOG 选项卡

Relaxation for feasOpt : 对目标的松弛量;

Mathematical programming->Preprocessing

Preprocessing aggregator application limit : 前处理聚合应用程序（主要是用于变量替换）的数量限制;

-1 Automatic (1 for LP, infinite for MIP)

0 Do not use any aggregator

Any positive integer

Bound strengthening indicator : 是否使用bound增强技术（仅对MIP有效）

-1 Automatic: let CPLEX choose

0 Do not apply bound strengthening

1 Apply bound strengthening



第八章 ILOG OPL进阶功能

ILOG 选项卡

Coefficient reduction setting : 是否使用系数缩减技术;

0 Do not use coefficient reduction

1 Reduce only to integral coefficients

2 Reduce all potential coefficients

Dependency indicator : 是否开启从属性检查;

-1 Automatic: let CPLEX choose

0 Off: do not use dependency checker

1 Turn on only at the beginning of preprocessing

2 Turn on only at the end of preprocessing

3 Turn on at the beginning and at the end of preprocessing



第八章 ILOG OPL进阶功能

ILOG 选项卡

Presolve dual setting : 是否进行对偶转换

-1 Off

0 Automatic: let CPLEX choose

1 On

Preprocessing aggregator fill : 变量替换的个数上限;

Linear reduction indicator : 是否只进行线性缩减;

0 Only linear reductions

1 Full reductions

Limit on the number of Presolve passes made : Presolve (用于简化和缩减问题) 的调用次数限制

-1 Determined automatically

0 Do not use Presolve; other reductions may still occur
or, any positive integer



信息科学与工程学院
COLLEGE OF INFORMATION SCIENCE AND ENGINEERING

第八章 ILOG OPL进阶功能

ILOG 选项卡

Presolve indicator : 是否启用Presolve

0 [CPX_OFF/false] Off (do not use presolve)

1 [CPX_ON/true] On (use presolve)

Indefinite MIQP indicator : 是否reformulate 只含0-1变量的 MIQP、MIQCP 0

[CPX_OFF/false] Off

1 [CPX_ON/true] On

Primal and dual reduction type : 确定基本和对偶缩减类型的设置

0 [CPX_PREREDUCE_NOPRIMALORDUAL]

No primal or dual reductions

1 [CPX_PREREDUCE_PRIMALONLY]

Only primal reductions

2 [CPX_PREREDUCE_DUALONLY]

Only dual reductions

3 [CPX_PREREDUCE_PRIMALANDDUAL]

Both primal and dual reductions



信息科学与工程学院
COLLEGE OF INFORMATION SCIENCE AND ENGINEERING

第八章 ILOG OPL进阶功能

ILOG 选项卡

Relaxed LP presolve indicator : 在MIP中根松弛是否采用presolve;

-1 Automatic: let CPLEX choose

0 [CPX_OFF] Off (do not use presolve on initial relaxation)

1 [CPX_ON] On (use presolve on initial relaxation)

Reapply presolve at root after processing : 处理根节点后是否重新采用presolve

-1 Automatic: let CPLEX choose (default)

0 Turn off represolve

1 Represolve without cuts

2 Represolve with cuts

3 Represolve with cuts and allow new root cuts



第八章 ILOG OPL进阶功能

ILOG 选项卡

Symmetry breaking : 在MIP预处理过程中是否采用symmetry breaking缩减;

-1 Automatic: let CPLEX determine the level of symmetry breaking

0 Off

1 CPLEX exerts a moderate level of symmetry breaking

2 CPLEX exerts an aggressive level of symmetry breaking

3 CPLEX exerts a very aggressive level of symmetry breaking

Simplex>Read

Scale parameter : 如何设定问题矩阵的比例

-1 No scaling

0 Equilibration scaling

1 More aggressive scaling

Data consistency checking indicator : 是否进行数据一致性检查

0 [CPX_OFF/false] Off (do not check)

1 [CPX_ON/true] On (check)



信息科学与工程学院
COLLEGE OF INFORMATION SCIENCE AND ENGINEERING

第八章 ILOG OPL进阶功能

ILOG 选项卡

Simplex->Simplex

Simplex crash ordering : CPLEX在选择初始基时, 如何确定变量次序;

Dual simplex pricing algorithm : 对偶单纯型的pricing (一种换基的策略) 算法选择;

0 [CPX_DPRIIND_AUTO] Automatic: let CPLEX choose

1 [CPX_DPRIIND_FULL] Standard dual pricing

2 [CPX_DPRIIND_STEEP] Steepest-edge pricing

3 [CPX_DPRIIND_FULL_STEEP]

Steepest-edge pricing in slack space

4 [CPX_DPRIIND_STEEPQSTART]

Steepest-edge pricing, unit initial norms

5 [CPX_DPRIIND_DEVEX] devex pricing



第八章 ILOG OPL进阶功能

ILOG 选项卡

Simplex iteration display information : CPLEX如何报告迭代信息;

0 No iteration messages until solution

1 Iteration info after each refactoring (重构)

2 Iteration info for each iteration

Simplex pricing candidate list size : 在pricing候选列表中的最大变量个数;

0 Determined automatically

or, any positive integer

Simplex refactoring frequency : Simplex基矩阵的重构频率;

0 Determined automatically

or, any integer from 1 to 10,000

Perturbation constant : CPLEX的上下扰动值;



第八章 ILOG OPL进阶功能

ILOG 选项卡

Simplex>Read

Simplex maximum iteration limit : 最大迭代次数;

Lower objective value limit : 目标值下限, 到了这个值算法就停止;

Simplex perturbation limit : Simplex的扰动上限

Simplex singularity repair limit : 求解如果遇到奇点时CPLEX试图修复的次数

Upper objective value limit : 目标值上限, 到了这个值算法也会停止;



第八章 ILOG OPL进阶功能

ILOG 选项卡

Simplex->tolerance

Feasibility tolerance : 可行性容差; 该值影响单纯型基的选择, 当优化不可行时, 可以尝试增加该值;

Markowitz tolerance : Markowitz容差; 该值影响单纯型基的选择;

Optimality tolerance : 最优性容差, 即影响最优解准则的reduced cost量 ;



第八章 ILOG OPL进阶功能

ILOG 选项卡

Mixed Integer Programming->General

MIP node log display information : MIP节点信息如何显示;

0 No display

1 Display integer feasible solutions

2 Display nodes under CPX_PARAM_MIPInterval

3 Same as 2 with information about node cuts

4 Same as 3 with LP subproblem information at root

5 Same as 4 with LP subproblem information at nodes

MIP node log interval : MIP节点记录日志的频率

MIP priority order generation : 设置MIP的一般优先级;

0 Do not generate a priority order

1 [CPX_MIPORDER_COST] Use decreasing cost

2 [CPX_MIPORDER_BOUNDS]

Use increasing bound range

3 [CPX_MIPORDER_SCALED COST]

Use increasing cost per coefficient count



信息科学与工程学院
COLLEGE OF INFORMATION SCIENCE AND ENGINEERING

第八章 ILOG OPL进阶功能

ILOG 选项卡

Mixed Integer Programming->Strategy

Backtracking tolerance : 回溯容差(0~1之间的数),控制分支过程中回溯的频率

MIP strategy best bound interval : 选择节点时是总选择最好的估计节点,还是间隔选取; 经验表明, 间隔选取效率高一些;

0 Best estimate node always selected
or, any positive integer

MIP branching direction : 算法的分支方向;

-1 [CPX_BRDIR_DOWN] Down branch selected first

0 [CPX_BRDIR_AUTO] Automatic: let CPLEX choose

1 [CPX_BRDIR_UP] Up branch selected first

MIP dive strategy : 算法的横向刺探策略;

0 Automatic: let CPLEX choose; 1 Traditional dive

2 Probing dive ; 3 Guided dive



信息科学与工程学院
COLLEGE OF INFORMATION SCIENCE AND ENGINEERING

第八章 ILOG OPL进阶功能

ILOG 选项卡

Node storage file indicator : 节点参数文件如何保存;

0 No node file

1 Node file in memory and compressed

2 Node file on disk

3 Node file on disk and compressed

MIP heuristic frequency : 设置启动周期性启发式算法的频率;

-1 None

0 Automatic: let CPLEX choose

or, any positive integer

Local branching heuristic : 是否启动本地分支启发式;



第八章 ILOG OPL进阶功能

ILOG 选项卡

MIP node selection strategy : 在回溯时如何选择下一个节点的策略;

0 [CPX_NODESEL_DFS] Depth-first search

1 [CPX_NODESEL_BESTBOUND] Best-bound search

2 [CPX_NODESEL_BESTEST] Best-estimate search

3 [CPX_NODESEL_BESTEST_ALT]

Alternative best-estimate search

MIP priority order generation : 是否启用MIP的一般优先级规则;

Node presolve selector : 是否在节点进行presolve;

-1 No node presolve

0 Automatic: let CPLEX choose

1 Force node presolve

2 Perform probing on integer-infeasible variables



第八章 ILOG OPL进阶功能

ILOG 选项卡

MIP probing level : 确定分支前横向探索的级别;

-1 No probing

0 Automatic: let CPLEX choose

1 Moderate probing level

2 Aggressive probing level

3 Very aggressive probing level

MIP starting algorithm : 最初MIP松弛为线性问题时优化器采用的算法;

0 [CPX_ALG_AUTOMATIC] Automatic

1 [CPX_ALG_PRIMAL] Primal Simplex

2 [CPX_ALG_DUAL] Dual Simplex

3 [CPX_ALG_NET] Network Simplex

4 [CPX_ALG_BARRIER] Barrier

5 [CPX_ALG_SIFTING] Sifting

6 [CPX_ALG_CONCURRENT] Concurrent Dual, Barrier and Primal



第八章 ILOG OPL进阶功能

ILOG 选项卡

MIP subproblem algorithm: 线性优化器遇到MIP子问题时采用的算法;

0 [CPX_ALG_AUTOMATIC] Automatic: let CPLEX choose

1 [CPX_ALG_PRIMAL] Primal Simplex

2 [CPX_ALG_DUAL] Dual Simplex

3 [CPX_ALG_NET] Network Simplex

4 [CPX_ALG_BARRIER] Barrier

5 [CPX_ALG_SIFTING] Sifting

MIP variable selection strategy : 分支结束后下一级分支变量的选取;

-1 [CPX_VARSEL_MININFEAS] Branch on variable with minimum infeasibility

0 [CPX_VARSEL_DEFAULT] Branch variable automatically selected

1 [CPX_VARSEL_MAXINFEAS] Branch on variable with maximum infeasibility

2 [CPX_VARSEL_PSEUDO] Branch based on pseudo costs

3 [CPX_VARSEL_STRONG] Strong branching

4 [CPX_VARSEL_PSEUDOREDUCED] Branch based on pseudo reduced costs



第八章 ILOG OPL进阶功能

ILOG 选项卡

Mixed Integer Programming->Limits

Constraint aggregation limit for cut generation : 在Cut时考虑多少个约束;

Number of cutting plane passes : 割平面的上限;

-1 None

0 Automatic: let CPLEX choose

Positive values give number of passes to perform

Row multiplier factor for cuts : cut的行乘因子, 控制cut的数量;

Candidate limit for generating Gomory fractional cuts : 产生Gomory cut的候选变量的上限个数;

MIP node limit : 节点个数上限;

Time spend polishing a solution: 修正解的时间上限(秒);



信息科学与工程学院
COLLEGE OF INFORMATION SCIENCE AND ENGINEERING

第八章 ILOG OPL进阶功能

ILOG 选项卡

Time spent probing : 横向探索的时间上限（秒）；

Try to repair infeasible MIP start : MIP启动时试图修复不可行解；

MIP solution limit: 解个数的上限；

MIP candidate list : 候选分支候选变量个数上限；

MIP simplex iterations : 在分支候选变量上进行的simplex迭代次数；

MIP parallel thread: 并行线程的个数；

Limit on nodes explored when a subMIP is being solved :
当解一个子MIP问题时节点的上限；

MIP thread limit: 线程上限；

Tree memory limit : branch & cut 树占用的内存上限值；



第八章 ILOG OPL进阶功能

ILOG 选项卡

Mixed Integer Programming->Tolerance

Absolute mipgap tolerance : 最好节点和最好整数解之间差的绝对容差;

Integrality tolerance : 整数解的容差 (在一定范围内仍被认为是整数);

Lower cutoff : 比此值小的解被剪枝;

Relative mipgap tolerance : 最好节点和最好整数解之间差的相对容差;

Absolute objective difference cutoff : 从当前最优解的目标值减去这个值 (最小化问题), 作为cutoff的值 (一般设为0);

Relative objective difference cutoff: 上述值的相对值;

Upper cutoff : 比此值大的解被剪枝;



第八章 ILOG OPL进阶功能

ILOG 选项卡

Mixed Integer Programming->Cuts

MIP cliques indicator : 是否启动clique（小范围） cuts;

-1 Do not generate clique cuts

0 Automatic: let CPLEX choose

1 Generate clique cuts moderately

2 Generate clique cuts aggressively

3 Generate clique cuts very aggressively

MIP covers indicator : 是否启动Cover cuts;

-1 Do not generate cover cuts

0 Automatic: let CPLEX choose

1 Generate cover cuts moderately

2 Generate cover cuts aggressively

3 Generate cover cuts very aggressively



第八章 ILOG OPL进阶功能

ILOG 选项卡

Mixed Integer Programming->Cuts

MIP disjunctive cuts indicator : 是否启用disjunctive cuts

- 1 Do not generate disjunctive cuts
- 0 Automatic: let CPLEX choose
- 1 Generate disjunctive cuts moderately
- 2 Generate disjunctive cuts aggressively
- 3 Generate disjunctive cuts very aggressively

MIP flow cover cuts indicator :是否启用flow cover cuts

- 1 Do not generate flow cover cuts
- 0 Automatic: let CPLEX choose
- 1 Generate flow cover cuts moderately
- 2 Generate flow cover cuts aggressively



第八章 ILOG OPL进阶功能

ILOG 选项卡

Mixed Integer Programming->Cuts

MIP Gomory fractional cuts indicator : 是否启用Gomory fractional cuts

- 1 Do not generate Gomory fractional cuts**
- 0 Automatic: let CPLEX choose**
- 1 Generate Gomory fractional cuts moderately**
- 2 Generate Gomory fractional cuts aggressively**

MIP GUB cuts indicator : 是否启用GUB cuts

- 1 Do not generate GUB cuts**
- 0 Automatically determined**
- 1 Generate GUB cuts moderately**
- 2 Generate GUB cuts aggressively**



第八章 ILOG OPL进阶功能

ILOG 选项卡

Mixed Integer Programming->Cuts

MIP implied bound cuts indicator : 是否启用 implied bound cuts

- 1 Do not generate implied bound cuts**
- 0 Automatically determined**
- 1 Generate implied bound cuts moderately**
- 2 Generate implied bound cuts aggressively**

MIP MIR (mixed integer rounding) cut indicator : 是否启用MIR cut;

- 1 Do not generate MIR cuts**
- 0 Automatic: let CPLEX choose**
- 1 Generate MIR cuts moderately**
- 2 Generate MIR cuts aggressively**



第八章 ILOG OPL进阶功能

ILOG 选项卡

Mixed Integer Programming->Cuts

MIP flow path cut indicator : 是否启用 **flow path cut** ;

-1 Do not generate flow path cuts

0 Automatic: let CPLEX choose

1 Generate flow path cuts moderately

2 Generate flow path cuts aggressively

Barrier->General

Barrier algorithm : 内点法使用的算法;

0 Default setting

1 Infeasibility-estimate start

2 Infeasibility-constant start

3 Standard barrier

注: **0**表示解决MIP子问题时用**1**, 其它情况用**3**;



信息科学与工程学院
COLLEGE OF INFORMATION SCIENCE AND ENGINEERING

第八章 ILOG OPL进阶功能

ILOG 选项卡

Barrier column nonzeros : 内点法dense columns 的数量;
0 Dynamically calculated
or, any positive integer

Convergence tolerance for LP and QP problems : LP和QP问题的收敛容差, 最小 $1e-12$;

Barrier crossover algorithm : 内点算法的后期是否使用交叉;
-1 No crossover
0 Automatic: let CPLEX choose
1 Primal crossover
2 Dual crossover

Barrier display information : 内点法显示信息的选项;
0 No progress information
1 Normal setup and iteration information
2 Diagnostic information



第八章 ILOG OPL进阶功能

ILOG 选项卡

Barrier ordering algorithm : 设定交换约束矩阵行时的排序算法;

0 Automatic: let CPLEX choose

1 Approximate minimum degree (AMD)

2 Approximate minimum fill (AMF)

3 Nested dissection (ND)

Convergence tolerance for quadratically constrained problems (QCP) : 二次约束问题的收敛容差, 最小 $1e-12$;

Barrier starting point algorithm : 设定计算内点法初始点的方法;

1 Dual is 0

2 Estimate dual

3 Average of primal estimate, dual 0

4 Average of primal estimate, estimate dual



第八章 ILOG OPL进阶功能

ILOG 选项卡

Barrier->Limits

Barrier maximum correction limit : 每次迭代时定位中心更正的最大量;

-1 Automatic: let CPLEX choose

0 None

or, any positive integer

Barrier growth limit : 如果问题没有无界最优面, 这个值选大较好, 缺省1e12;

Barrier iteration limit : 迭代上限;

Barrier objective range: 目标值上限;

Barrier thread limit: 算法线程的上限;



第八章 ILOG OPL进阶功能

ILOG 选项卡

Network->General

Network logging display indicator :设定网络日志的显示;

0 [CPXNET_NO_DISPLAY_OBJECTIVE] No display

1 [CPXNET_TRUE_OBJECTIVE]

Display true objective values

2 [CPXNET_PENALIZE_OBJECTIVE]

Display penalized objective values

Network simplex iteration limit : 网络单纯型的迭代上限;

Simplex network extraction level : 网络单纯型的网络提取级别

1 [CPX_NETFIND_PURE] Extract pure network only

2 [CPX_NETFIND_REFLECT] Try reflection scaling

3 [CPX_NETFIND_SCALE] Try general scaling



第八章 ILOG OPL进阶功能

ILOG 选项卡

Network Simplex pricing algorithm : 网络单纯型的pricing算法选择;

0 [CPXNET_PRICE_AUTO] Automatic: let CPLEX choose

1 [CPXNET_PRICE_PARTIAL] Partial pricing

2 [CPXNET_PRICE_MULT_PART] Multiple partial pricing

3 [CPXNET_PRICE_SORT_MULT_PART] Multiple partial pricing with sorting

Network->Tolerance

Feasibility tolerance for CPXNETprimopt : 网络单纯型的可行性容差;

Optimality tolerance for CPXNETprimopt : 网络单纯型的最优性容差;



第八章 ILOG OPL进阶功能

ILOG 选项卡

Sifting->General

Sifting display information : 筛选法的显示设置;

0 No display

1 Display major iterations

2 Display LP subproblem information within each sifting iteration

Sifting subproblem algorithm : 筛选法子问题的算法;

0 [CPX_ALG_AUTOMATIC] Automatic: let CPLEX choose

1 [CPX_ALG_PRIMAL] Primal Simplex

2 [CPX_ALG_DUAL] Dual Simplex

3 [CPX_ALG_NET] Network Simplex

4 [CPX_ALG_BARRIER] Barrier

Upper limit for sifting iteration:筛选法迭代上限;

