ICMC USP SÃO CARLOS

Programação por restrições

Luiz Henrique Cherri
Thiago Queiroz

Escopo

1. Otimização
2. O paradigma de programação por restrições
3. Modelando problemas utilizando programação por restrições
4. Exercício
5. Resolvedores de programação por restrições

Otimização

O que é otimização

- No dicionário OTIMIZAR é
  - “Ocasionar circunstâncias mais proveitosas para; retirar o que há de melhor em; aprimorar, melhorar: otimizar o desenvolvimento do produto; otimizar as condições de trabalho.”
  - “Otimização é o processo de otimizar, de tornar ótimo. É a busca da excelência. É o emprego de técnicas para seleção das melhores alternativas, com o propósito de alcançar os objetivos determinados.”

- Muitas, muitas aplicações
  - Problema de corte, designação, mistura, roteamento de veículos
- Propriedades
  - Computacionalmente difícil
  - São necessárias técnicas de modelagem e experiência
  - Natureza experimental
  - Importante na pratica (R$)
- Muitas estratégias de solução
  - Modelagem e resolução via métodos genéricos
  - Métodos especializados
  - Busca local e metaheuristicas

Como resolver um problema de otimização

- Métodos heurísticos
  - Obtenção de soluções de boa qualidade
  - Sem garantia de otimalidade
  - Tempo computacional baixo

- Métodos exatos
  - Garantia de otimalidade
  - Há uma medida de qualidade para a solução
  - Tempo computacional elevado

Como modelar um problema?

Problema real

Simplificações no problema real

Representação por variáveis e restrições

Validação do modelo

Tipos de modelagem

Programaçao linear

Programação inteira e inteira mista

Programação não-linear

Programação dinâmica

Programação por restrições

Mas, porque essas classes são importantes?

- Porque podem ser desenvolvidos métodos genéricos que:
  - Tomam proveito das características do problema
  - Podem ser aplicados genericamente a qualquer problema da classe

- A priori, não há forma certa ou errada de se modelar um problema, mas, sabendo das características de cada tipo de modelagem, pode-se decidir as aproximações que serão utilizadas

- Por exemplo...

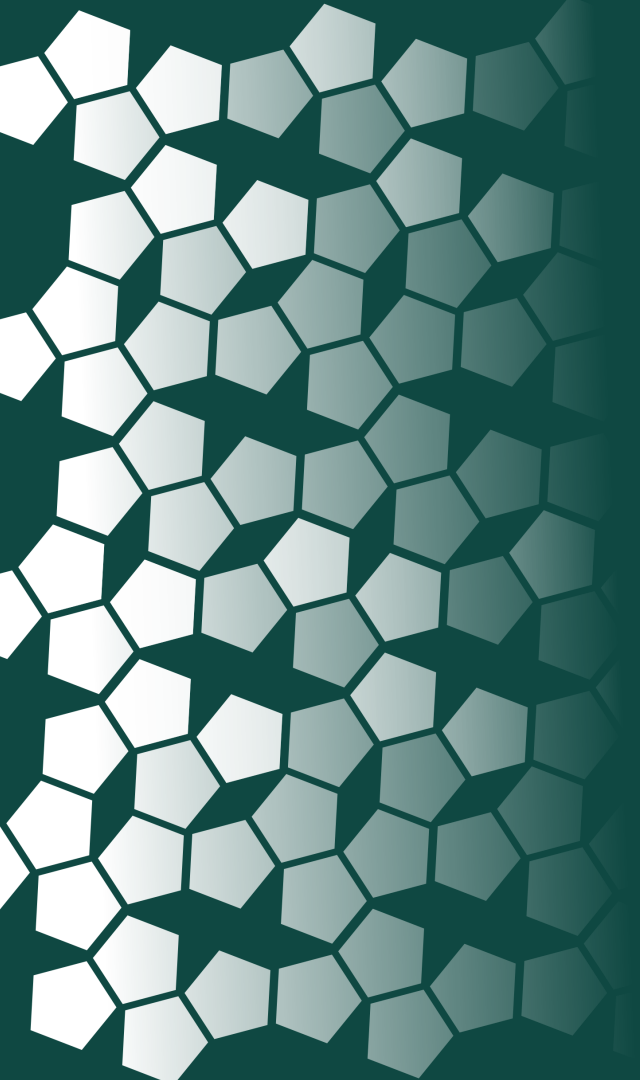
CP ou MIP

- Branch & Prune
  - Prune: elimina configurações infactíveis
  - Branch: decomposição em subproblemas
- Prune
  - Examina as restrições para reduzir os possíveis valores das variáveis
- Branch
  - Usa heurísticas baseadas na informação de factibilidade
- Foco: restrições e factibilidade

- Branch & Bound
  - Bound: elimina soluções sub-ótimas
  - Branch: decomposição em subproblemas
- Bound
  - Utiliza uma relaxação do problema eliminar nós da busca
- Branch
  - Usa a informação da relaxação
- Foco: função objetivo e otimalidade

Quem já usou?

- Diversos autores já utilizaram programação por restrições com sucesso
- Exemplos podem ser encontrados em:
  - Refalo (1999): de 2 a 200 vezes mais rápido que programação inteira
  - Hooker & Osorio (1999): de 4 a 150 vezes mais rápido que programação inteira
  - Refalo (1999): de 2 a 200 vezes mais rápido que programação inteira
  - Thorsteinsson & Ottosson (2001): de 30 a 40 vezes mais rápido que programação inteira



O paradigma de programação por restrições

O paradigma de programação por restrições

- Basicamente, a ideia da programação por restrições é:
  - Declarar restrições sobre as variáveis do problema
  - Encontra uma solução que satisfaça todas as restrições

- Uma restrição é uma relação entre as variáveis, por exemplo,
  - $x < y$
  - $x + y = z - 1$
  - $(x^3 \div y) = z$

O paradigma de programação por restrições

- Uma restrição é uma relação entre as variáveis que também pode ser expressa por condições lógicas
- Existem várias condições lógicas já previamente definidas
  - IFTHEN IF (x = 1) THEN (y != 0)
  - COUNT COUNT (X = 1)
  - ALLDIFF ALLDIFF (X)
  - MAX MAX (X)
  - MIN MIN (X)
  - … Muitas outras

Especificamente

- Variáveis e domínios
  - As variáveis devem possuir domínios finito, i.e.
    - $X = \{1,2,3\}$
    - $K = \{7,22,34,99\}$
    - $M = \{0.1,0.3,1.7\}$
- Restrições
  - Devem criar uma relação entre as variáveis, representando o problema real
  - A modelagem pode ser feita de inúmeras maneiras
    - Deve se pensar em restrições que sejam eficientes na redução dos domínios das variáveis
- Uma solução factível é obtida quando a cada variável for atribuído um único valor e, estes valores respeitam todas as restrições

Exemplo: coloração de mapas

- Colorir o mapa de parte da Europa: Bélgica, Suíça, França, Alemanha, Holanda, Luxemburgo
- Os países adjacentes não podem possuir a mesma cor
- Quatro cores (azul, vermelho, amarelo, e cinza) seriam suficientes?

Exemplo: coloração de mapas

- O que podem ser nossas variáveis
  - Paises = {Belgica, Suica, Franca, Alemanha, Holanda, Luxemburgo}
- Qual o domínio de cada variável?
  - {azul, vermelho, amarelo, cinza}
- Como representar estas restrições
  - Belgica != Luxemburgo
  - Belgica != Franca
  - Belgica != Alemanha
  - Belgica != Holanda
  - Suica != Franca
  - … e assim por diante

Exemplo: coloração de mapas

Note que:

- Variáveis são não-numéricas
- Restrições utilizadas não são lineares
- Não se parece com um problema de programação inteira!
- É perfeitamente exequível utilizando programação por restrições

Então precisamos definir:

- Domínios
  - Para cada variável: qual é o conjunto de valores possíveis?
  - Se vazio para qualquer variável, então a solução é inviável
  - Se único para qualquer variável, então, solução parcial

- Restrições
  - Capturar subestruturas interessantes e bem estudados
  - É preciso:
    - Determinar se a restrição é viável de acordo com o domínio das variáveis
    - Podar valores "impossíveis" dos domínios

Exemplo: Análise para a redução de domínios

Pode haver diferentes técnicas para lidar com cada tipo de restrição
Por exemplo:

Restrição: $3x + 10y + 2z + 4w = 4$
Domínios: $x \in \{0,1\}$, $y \in \{0,1,2\}$, $z \in \{0,1,2\}$, $w \in \{0,1\}$

Analisado a equação, temos que y deve ter seu domínio reduzido a {0}
Com uma análise mais detalhada temos:

x tem domínio {0}, y tem domínio {0},
z tem domínio {0,2}, w tem domínio {0,1}

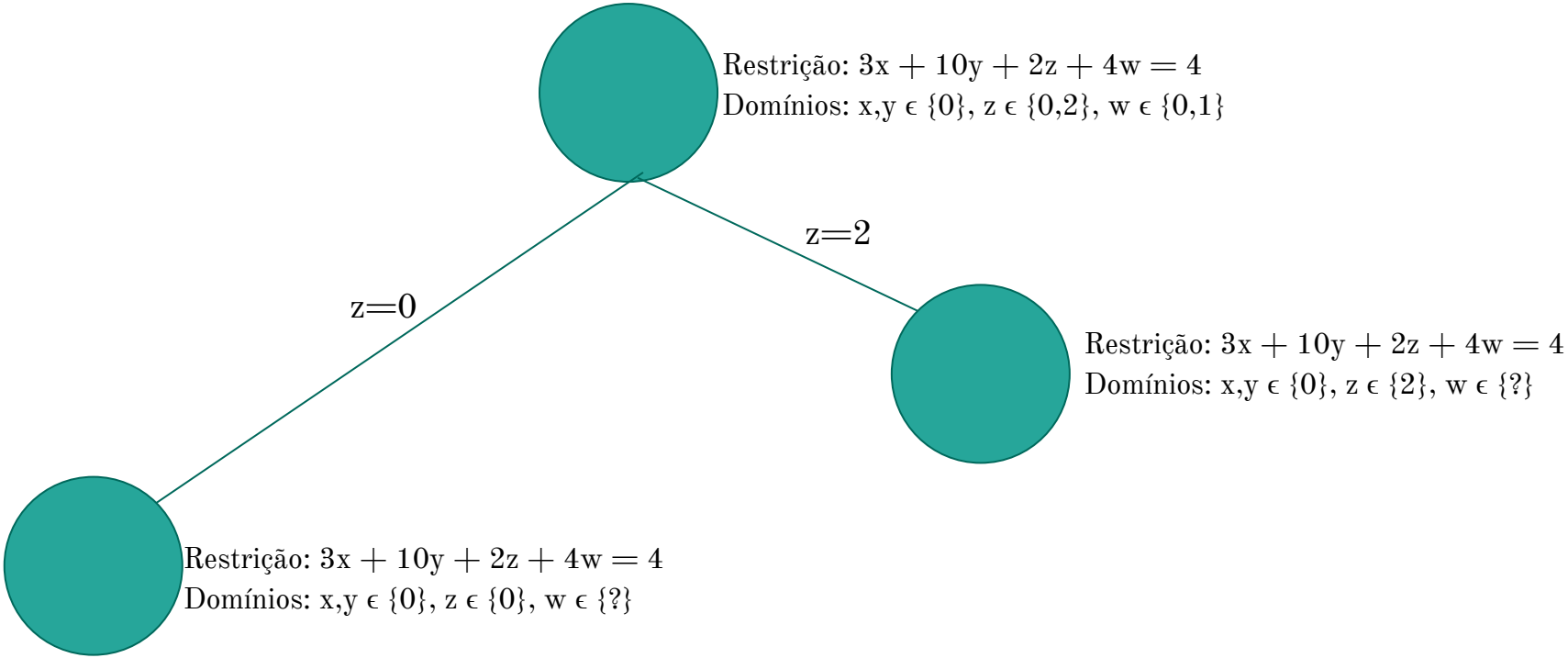
Exemplo: Análise para a redução de domínios

- Este processo de redução de domínios é chamado de propagação
- A propagação pode reduzir muito o conjunto de valores possíveis para as variáveis
- Para encontrar uma solução factível temos se utiliza uma árvore de busca

Temos o problema com os domínios já propagados

Restrição: $3x + 10y + 2z + 4w = 4$
Domínios: $x \in \{0\}$, $y \in \{0\}$, $z \in \{0,2\}$, $w \in \{0,1\}$

Ramificando



Ou seja...

O método de solução é bastante similar ao *Branch & Bound*

O que muda é:

- Ao invés de relaxação, temos a propagação/redução de domínios
- Quanto mais esta propagação reduzir o domínio de busca, menor será a árvore de busca

Perguntas

Quando ramificamos o valor de uma variável na árvore, o que esperamos?

Quando a busca acaba?

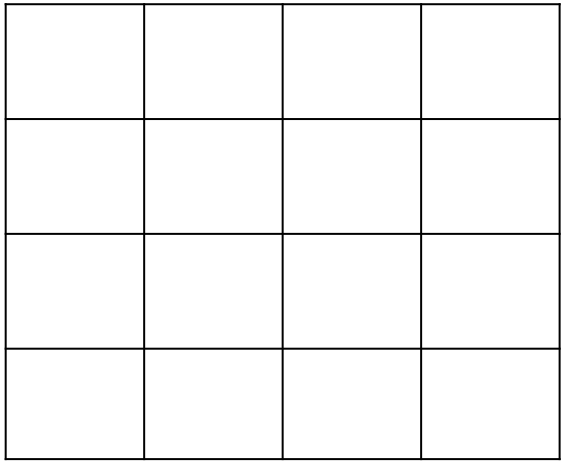
Quais as vantagens de não resolver uma relaxação a cada nó?

Quais as características de um problema que o torna eficiente de ser resolvido via programação por restrições?

Exemplo

4-rainhas

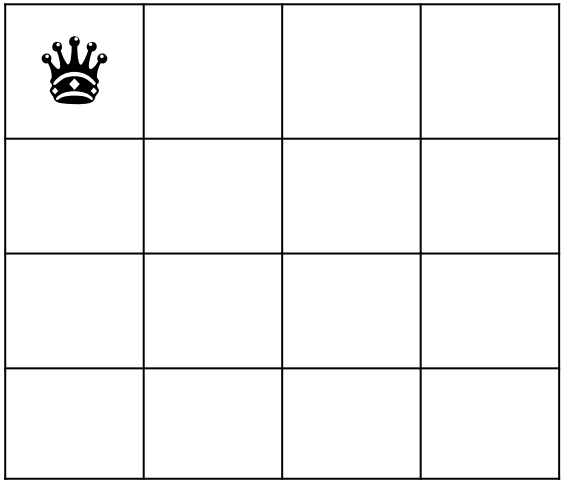
- As rainhas de um tabuleiro de xadrez podem atacar na vertical, horizontal e diagonal
- Como posicionar 4 rainhas no tabuleiro 4x4 sem que um par de rainhas possam se atacar?
- Como deve ser a estrutura da solução para o problema?



4-rainhas

Arbitrariamente posicionamos uma rainha no canto superior esquerdo,

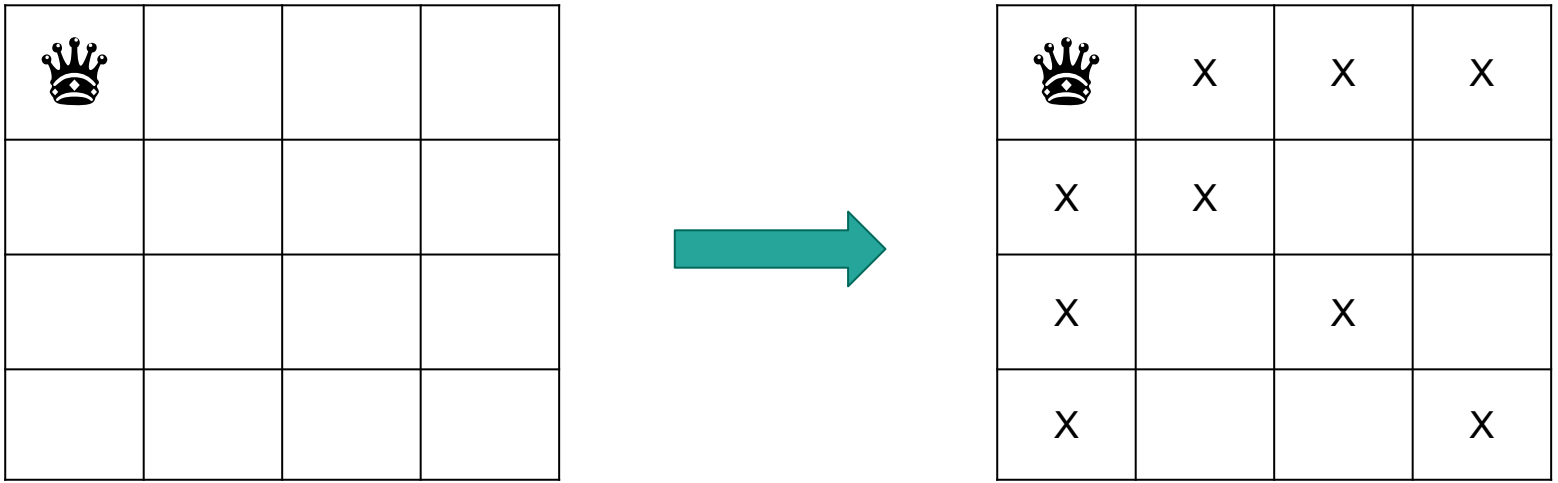
O que ocorre na propagação?



4-rainhas

Arbitrariamente posicionamos uma rainha no canto superior esquerdo,

O que ocorre na propagação?



4-rainhas

Novamente, uma rainha em uma posição arbitrária e propagamos,

O que ocorre na propagação?

♕	X	X	X
X	X		
X	♕	X	
X			X

4-rainhas

Novamente, uma rainha em uma posição arbitrária e propagamos,

O que ocorre na propagação?

♛	X	X	X
X	X		
X	♛	X	
X			X

♛	X	X	X
X	X	**X**	
X	♛	X	**X**
X	**X**	**X**	X

4-rainhas

Esta solução pode levar a no máximo 3 rainhas no tabuleiro…

O que fazer?

♕	X	X	X
X	X	**X**	
X	♕	X	**X**
X	**X**	**X**	X

4-rainhas - Backtracking

Novamente, uma rainha em uma posição arbitrária e propagamos,

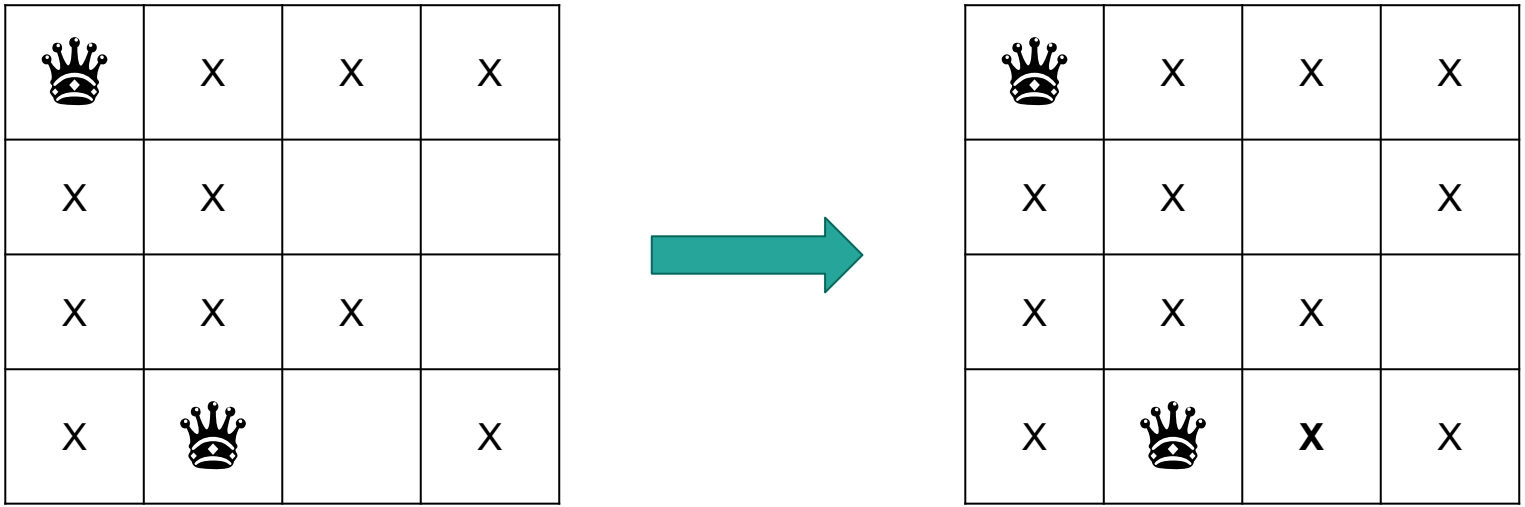
O que ocorre na propagação?

♕	x	x	x
x	x		
x	x	x	
x	♕		x

4-rainhas - Backtracking

Novamente, uma rainha em uma posição arbitrária e propagamos,

O que ocorre na propagação?

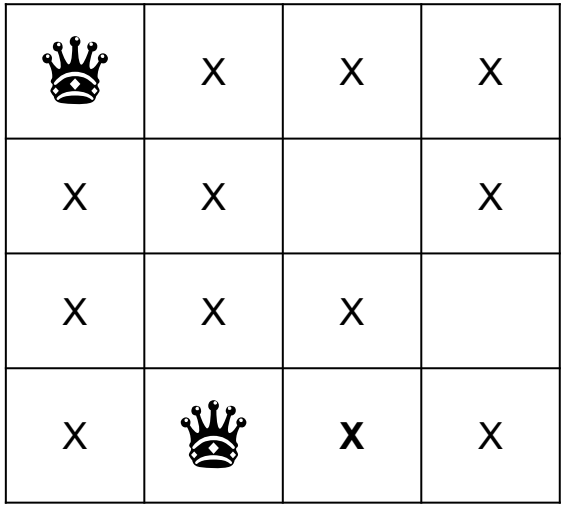


4-rainhas

Novamente, Esta configuração nos levará a apenas 3 rainhas no tabuleiro (porque?)

Efetuamos o backtracking

Qual rainha devemos mudar de posição?



4-rainhas

Mudamos a posição da primeira rainha

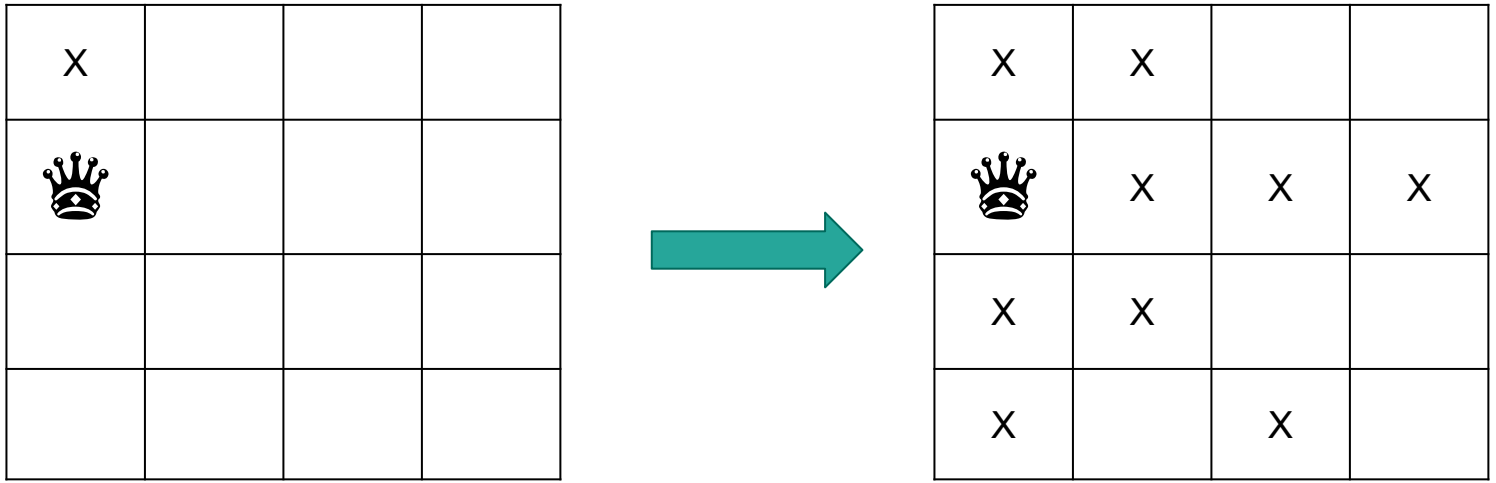
O que ocorre na propagação?

X			
♕			

4-rainhas

Mudamos a posição da primeira rainha na coluna

O que ocorre na propagação?



4-rainhas

Alocamos uma rainha na segunda coluna (na única posição viável)

O que ocorre na propagação?

X	X		
♕	X	X	X
X	X		
X	♕	X	

4-rainhas

Alocamos uma rainha na segunda coluna

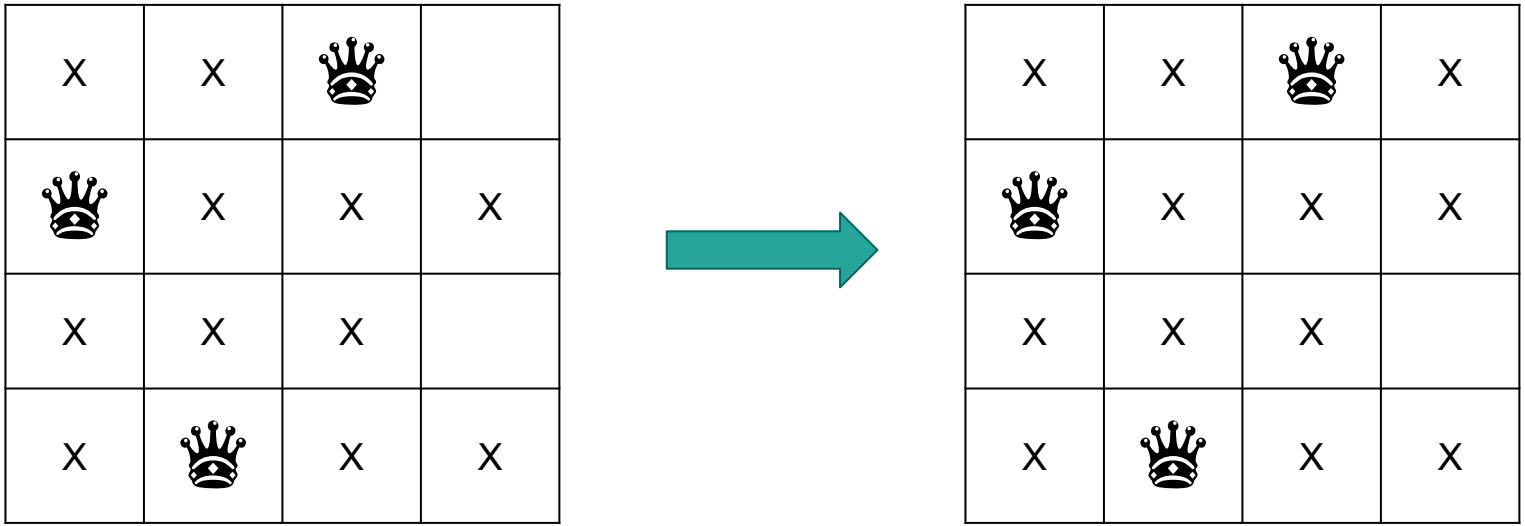
O que ocorre na propagação?

X	X		
♛	X	X	X
X	X		
X	♛	X	

X	X		
♛	X	X	X
X	X	X	
X	♛	X	X

4-rainhas

Alocamos uma rainha na terceira coluna e propagamos



4-rainhas

Assim, obtemos a solução. Ela é única?

X	X	♕	X
♕	X	X	X
X	X	X	♕
X	♕	X	X

Modelando

Mas… e se o problema 4-queens fosse 12-queens…

O procedimento é o mesmo mas, resolver manualmente o problema pode ser uma tarefa difícil.

Modelando o problema (Exercício)

- Como podemos definir as variáveis?
- Quais restrições garantem que serão satisfeitas as condições do problema

Resolução - variáveis

Criamos um vetor com 12 variáveis

Q =

1	2	3	4	5	6	7	8	9	10	11	12

O domínio de cada variável é {1,2,3,4,5,6,7,8,9,10,11,12} (números de 1 à 12)

Resolução - variáveis

Criamos um vetor com 12 variáveis

Q =

1	2	3	4	5	6	7	8	9	10	11	12

O domínio de cada variável é {1,2,3,4,5,6,7,8,9,10,11,12} (números de 1 à 12)

- O valor de cada variável indica a coluna uma das rainhas é alocada
- A posição desta variável no vetor indica a linha que a rainha é alocada

Resolução

Q =

1	2	3	4	5	6	7	8	9	10	11	12

Assim

- Não haverá duas rainhas na mesma linha (porque?)

Resolução

Para garantir que não existam duas rainhas na mesma coluna,

AllDifferent (Q[i] , i=1,...,12)

Assegurando assim que todas as rainhas são alocadas em colunas diferentes

Mas, como assegurar que não há rainhas na mesma diagonal???

Resolução

Não é tão complicado quanto parece

Precisamos somente das seguintes restrições

- AllDifferent (Q[i] + i , i=1,...,12)
- AllDifferent (Q[i] − i , i=1,...,12)

Isso funciona? Façam um teste aleatório.

Por que funciona?

Restrições globais

Restrições globais

Você pode propor sua própria restrição!!!

Por que fariamos isso?

Restrições globais

Você pode propor sua própria restrição!!!

Por que fariamos isso?

Aproveitar uma estrutura particular do problema

Eliminar caminhos que não irão levar a uma solução viável logo no início da busca

Reduzir o número de operações para inferir domínios

Reduzir o tempo para a exploração das soluções

Exemplo de restrições globais

- N. Beldiceanu, M. Carlsson, (2002), *A New Multi-Resource cumulatives Constraint with Negative Heights*
- C. Bessière, E. Hebrard, B. Hnich, Z. Kızıltan, T. Walsh, *The range and roots Constraints: Specifying Counting and Occurrence Problems*
- A. M. Frisch, B. Hnich, Z. Kızıltan, I. Miguel, T. Walsh, Global Constraints for Lexicographic Orderings

E muitas outras

- Catalogo: http://sofdem.github.io/gccat/

Exercício

Exercício - SUDOKU

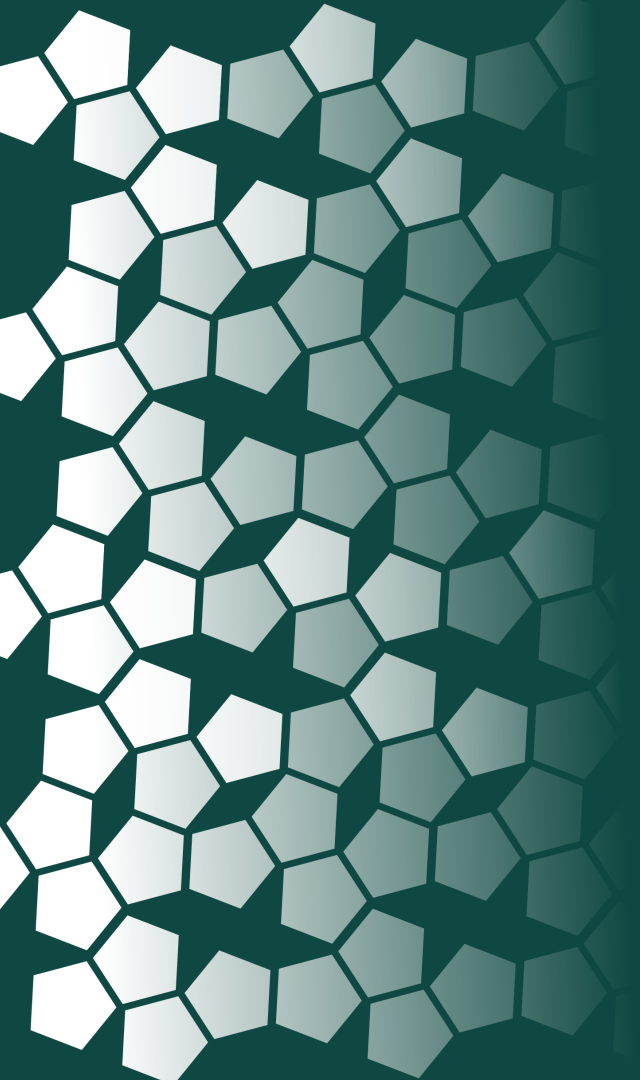
- Completar a matriz ao lado com elementos de 1 a 9
- Não se pode repetir elementos:
  - Nas linhas
  - Nas colunas
  - Nos sub blocos 3x3 indicados

		5	3					
8							2	
	7			1		5		
4					5	3		
	1			7				6
		3	2				8	
	6		5					9
		4					3	
					9	7		

Exercício - SUDOKU

- Completar a matriz ao lado com elementos de 1 a 9
- Não se pode repetir elementos:
  - Nas linhas
  - Nas colunas
  - Nos sub blocos 3x3 indicados
- Quais são as variáveis?
- Quais restrições utilizar?

		5	3					
8							2	
	7			1		5		
4					5	3		
	1			7				6
		3	2				8	
	6		5					9
		4					3	
					9	7		



Modelando um problema com programação por restrições

Solvers de programação por restrições

- Comerciais
  - IBM ILOG CP
  - SCIP*
- Livres
  - Google or/tools
  - Choco solver
  - Baseados em PROLOG (como o ‘CHIP’)
  - SCIP*
- Muitos outros (www.constraintsolving.com/solvers)

IBM ILOG CP

- Software comercial
- Livre para uso acadêmico
- Para fins do curso, a versão de avaliação é suficiente!
  - Poucas variáveis
  - Poucas restrições

- Link para cadastro e download:
  https://www-01.ibm.com/software/websphere/products/optimization/cplex-studio-community-edition

Baixar o CPLEX community edition

Acessando

Acesso via windows:

- Menu do sistema > oplide

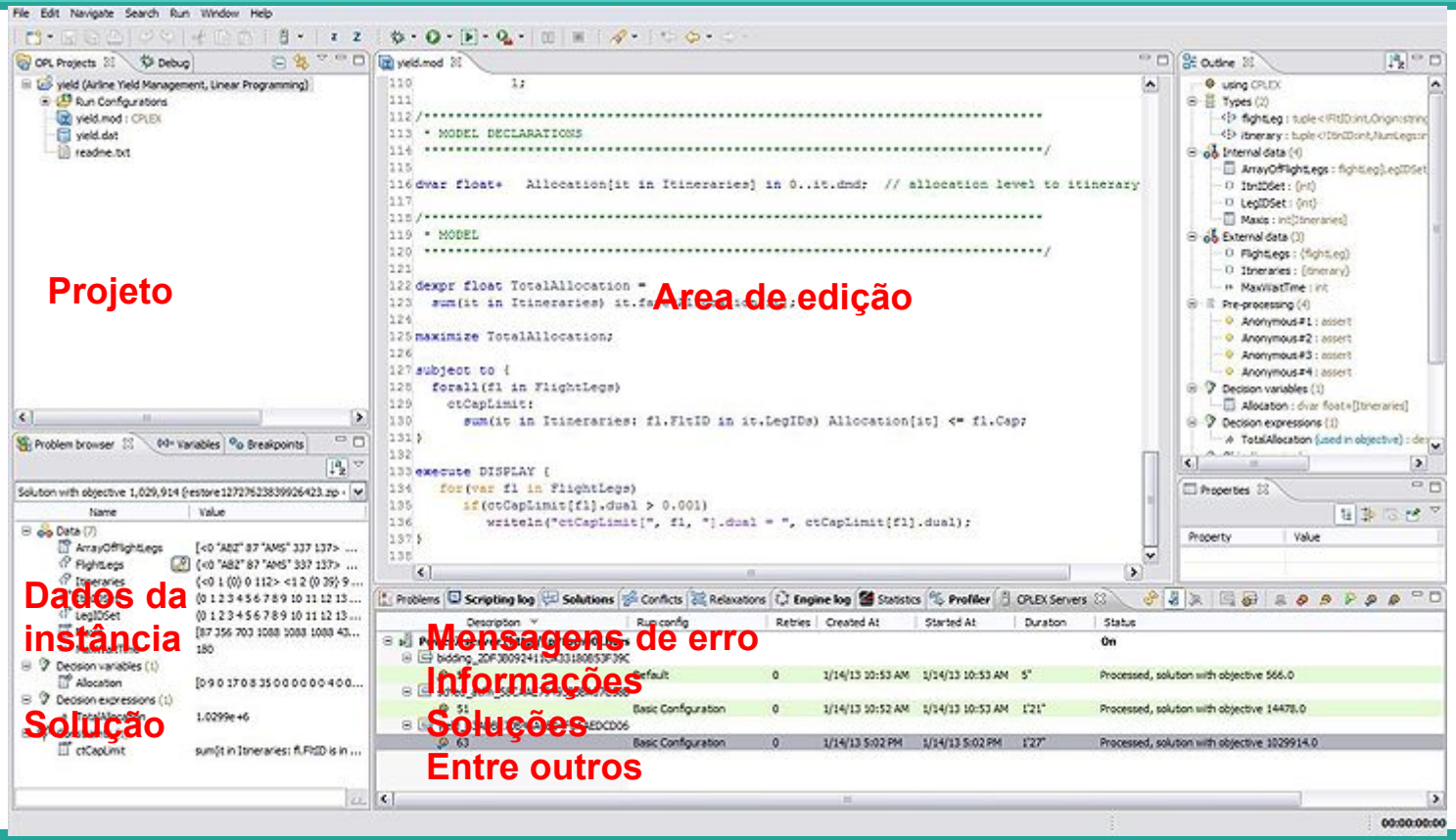
Acesso via Linux (ubuntu):

- No terminal:
  - cd /opt/ibm/ILOG/CPLEX_<VERSION>/opl/oplide/
  - ./oplide

Acesso via OSx: O CPLEX não disponibiliza a interface para o sistema!

Abrindo temos...

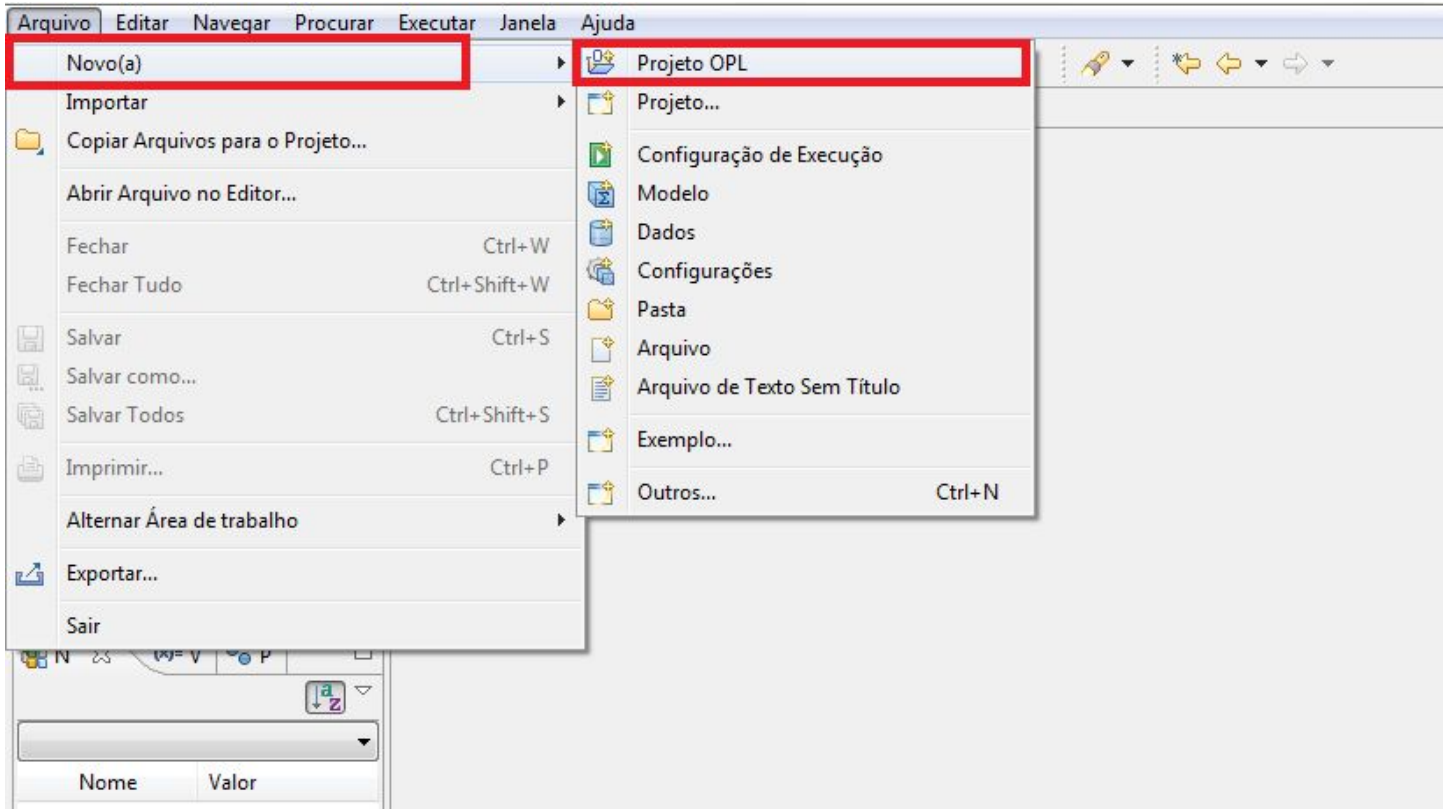
Interface

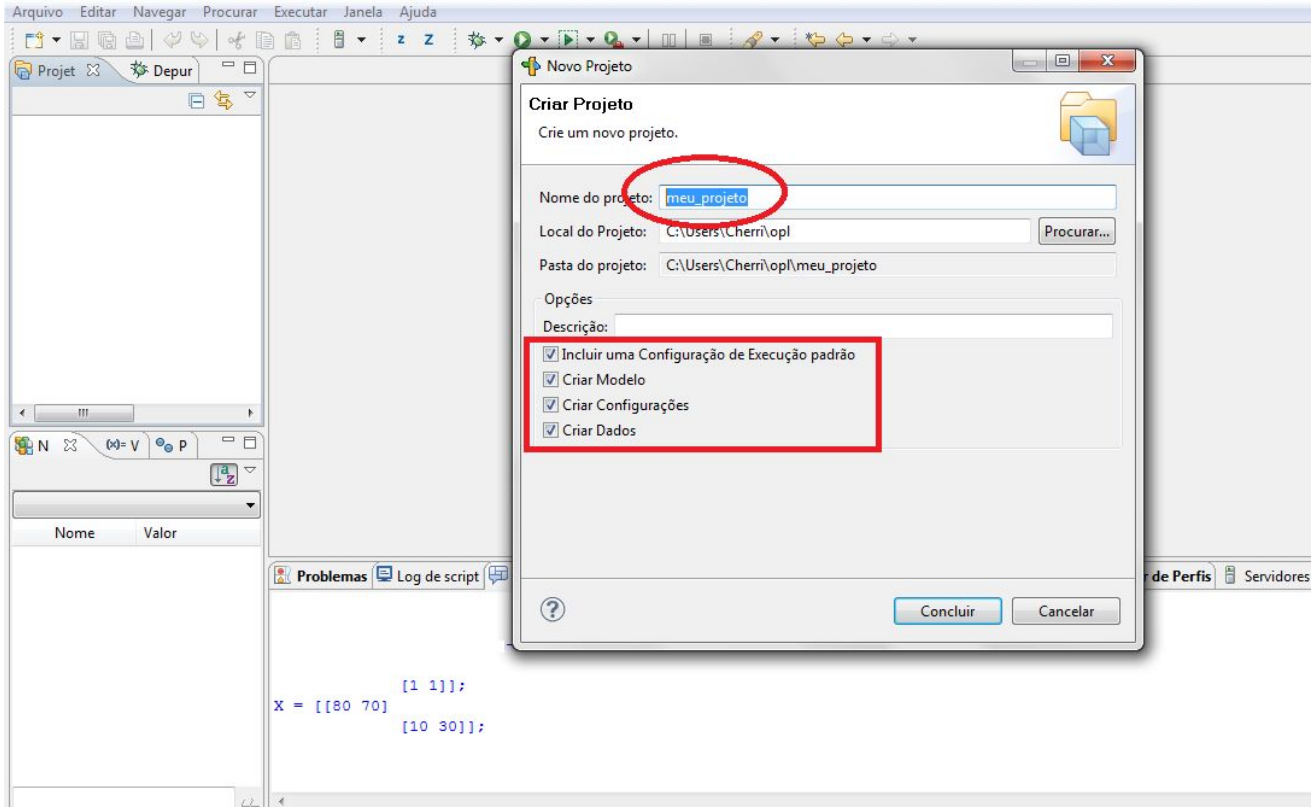

Projeto
Area de edição
Dados da instância
Solução
Mensagens de erro
Informações
Soluções
Entre outros

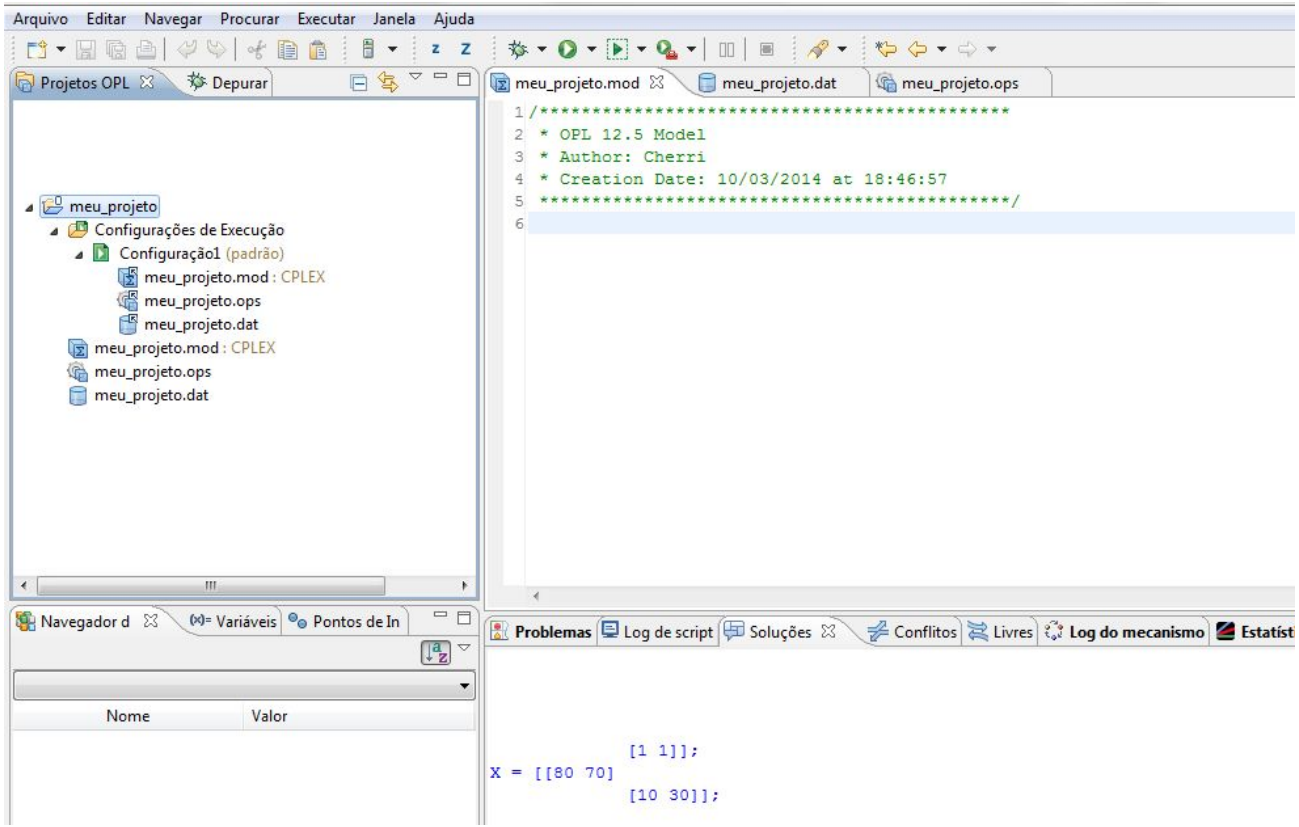
Componentes básicas de um projeto

- Um projeto pode ser constituído por três tipos de arquivos:
  - .mod: onde o modelo é implementado;
  - .dat: dados usados na criação do modelo;
  - .ops: arquivo de configuração.

- Para se relacionarem, esses arquivos devem estar ligados à uma mesma componente de execução.


Arquivo
Editar
Navegar
Procurar
Executar
Janela
Ajuda
Novo(a)
Importar
Copiar Arquivos para o Projeto...
Abrir Arquivo no Editor...
Fechar Ctrl+W
Fechar Tudo Ctrl+Shift+W
Salvar Ctrl+S
Salvar como...
Salvar Todos Ctrl+Shift+S
Imprimir... Ctrl+P
Alternar Área de trabalho
Exportar...
Sair
Projeto OPL
Projeto...
Configuração de Execução
Modelo
Dados
Configurações
Pasta
Arquivo
Arquivo de Texto Sem Título
Exemplo...
Outros... Ctrl+N
Nome
Valor


Arquivo Editar Navegar Procurar Executar Janela Ajuda
Novo Projeto
Criar Projeto
Crie um novo projeto.
Nome do projeto: meu_projeto
Local do Projeto: C:\Users\Cherri\opl
Procurar...
Pasta do projeto: C:\Users\Cherri\opl\meu_projeto
Opções
Descrição:
Incluir uma Configuração de Execução padrão
Criar Modelo
Criar Configurações
Criar Dados
Concluir
Cancelar
Problemas
Log de script
Nome
Valor
[1 1]];
X = [[80 70]
[10 30]];


Arquivo Editar Navegar Procurar Executar Janela Ajuda
Projetos OPL
Depurar
meu_projeto
Configurações de Execução
Configuração1 (padrão)
meu_projeto.mod : CPLEX
meu_projeto.ops
meu_projeto.dat
meu_projeto.mod : CPLEX
meu_projeto.ops
meu_projeto.dat
meu_projeto.mod
meu_projeto.dat
meu_projeto.ops
1 /***
2 * OPL 12.5 Model
3 * Author: Cherri
4 * Creation Date: 10/03/2014 at 18:46:57
5 ***/
6
Navegador d
Variáveis
Pontos de In
Nome
Valor
Problemas
Log de script
Soluções
Conflitos
Livres
Log do mecanismo
Estatíst
[1 1]];
X = [[80 70]
[10 30]];

OBRIGADO!