PICAT: Uma Linguagem de Programação Multiparadigma

Miguel Alfredo Nunes, Jeferson L. R. Souza, Claudio Cesar de Sá

miguel.nunes@edu.udesc.br

jeferson.souza@udesc.br

claudio.sa@udesc.br

Departamento de Ciência da Computação
Centro de Ciências e Tecnológias
Universidade do Estado de Santa Catarina

16 de abril de 2019

Contribuições

- Alexandre Gonçalves;
- João Herique Faes Battisti;
- Paulo Victor de Aguiar;
- Rogério Eduardo da Silva;
- Hakan Kjellerstrand – (`http://www.hakank.org/picat/`)
- Neng-Fa Zhou – (`http://www.picat-lang.org/`)
- Outros anônimos que auxiliaram na produção deste documento;

Apresentação ao Curso de PICAT – I

- O que é o PICAT?

Apresentação ao Curso de PICAT – I

- O que é o PICAT?
  - Uma linguagem de programação de propósitos gerais
  - Uma evolução do PROLOG (consagrada linguagem dos primórdios da IA)
  - Tem elementos das linguagens Python, Prolog e Haskell
- Uso e finalidades do PICAT:

Apresentação ao Curso de PICAT – I

- O que é o PICAT?
  - Uma linguagem de programação de propósitos gerais
  - Uma evolução do PROLOG (consagrada linguagem dos primórdios da IA)
  - Tem elementos das linguagens Python, Prolog e Haskell
- Uso e finalidades do PICAT:
  - Uso de programas gerais: de simples à complexos (uma reflexão)
  - Provê suporte há vários *solvers* na área de Pesquisa Operacional
  - Área: IA, programação por restrições, programação inteira, planejamento, combinatória, etc

Apresentação ao Curso de PICAT – II

- Este curso é dirigido a voce?

Apresentação ao Curso de PICAT – II

- Este curso é dirigido a voce?
- Requisitos:

Apresentação ao Curso de PICAT – II

- Este curso é dirigido a voce?
- Requisitos:
  - Conhecimento: noções de lógica matemática (proposional e primeira-ordem), matemática elementar, e alguma outra linguagem de programação
  - Dedicação: depende de você

Apresentação ao Curso de PICAT – II

- Este curso é dirigido a voce?
- Requisitos:
  - Conhecimento: noções de lógica matemática (proposional e primeira-ordem), matemática elementar, e alguma outra linguagem de programação
  - Dedicação: depende de você
- Motivação:

Apresentação ao Curso de PICAT – II

- Este curso é dirigido a voce?
- Requisitos:
  - Conhecimento: noções de lógica matemática (proposional e primeira-ordem), matemática elementar, e alguma outra linguagem de programação
  - Dedicação: depende de você
- Motivação:
  - Dependendo de sua dedicação, ao final voce vai estar apto a resolver problemas computacionais de simples à difíceis
  - Difícil: muitas linhas de código e muito conhecimento de algoritmos seriam necessários
  - Com Picat, há sofisticados esquemas prontos para se construir programas.

Apresentação ao Curso de PICAT – III

- Requisitos computacionais:

Apresentação ao Curso de PICAT – III

- Requisitos computacionais: Um computador qualquer (arquitetura 16, 32 ou 64 bits), com Linux, Mac ou Windows, que tenha um compilador C instalado completo, preferencialmente.
- Comunidade e ações: `http://picat-lang.org`

Apresentação ao Curso de PICAT – III

- Requisitos computacionais: Um computador qualquer (arquitetura 16, 32 ou 64 bits), com Linux, Mac ou Windows, que tenha um compilador C instalado completo, preferencialmente.
- Comunidade e ações: `http://picat-lang.org`
- Códigos e este material, sempre atualizados em:

Apresentação ao Curso de PICAT – III

- Requisitos computacionais: Um computador qualquer (arquitetura 16, 32 ou 64 bits), com Linux, Mac ou Windows, que tenha um compilador C instalado completo, preferencialmente.
- Comunidade e ações: `http://picat-lang.org`
- Códigos e este material, sempre atualizados em:
  - Este PDF e seu texto original: `http://github.com/claudiosa/Slides_Picat`
  - Os códigos de programas: `http://github.com/claudiosa/CCS/picat`
- Além do material aqui disponível em PDF, o mais importante do curso vai estar na interatividade da minha **apresentação oral**.
- Ou seja, este material é um guia para o seu desenvolvimento, mas minhas explicações são horas de estudo e código feito

Apresentação ao Curso de PICAT – IV

- Além desta apresentação do curso, voce pode assistir uma parte deste curso em aulas que fiz para o Youtube, há alguns anos atrás:

Apresentação ao Curso de PICAT – IV

- Além desta apresentação do curso, voce pode assistir uma parte deste curso em aulas que fiz para o Youtube, há alguns anos atrás:
- Videoaula 01: Introdução ao PICAT
  `https://www.youtube.com/watch?v=0DmTyFFQPK8`

Apresentação ao Curso de PICAT – IV

- Além desta apresentação do curso, voce pode assistir uma parte deste curso em aulas que fiz para o Youtube, há alguns anos atrás:
- Videoaula 01: Introdução ao PICAT
  `https://www.youtube.com/watch?v=0DmTyFFQPK8`
- Videoaula 02: Tipos de Dados do PICAT
  `https://www.youtube.com/watch?v=7fPKPd0ZDnc`
- Estas videoaulas forem refeitas e detalhadas neste curso. Aqui encontram-se com uma outra abordagem.

Apresentação ao Curso de PICAT – V

- Assim, ao final deste curso terás uma visão forte de uma ferramenta computacional, utilizada em várias áreas tais como: modelagem matemática, IA, Pesquisa Operacional, etc

Apresentação ao Curso de PICAT – V

- Assim, ao final deste curso terás uma visão forte de uma ferramenta computacional, utilizada em várias áreas tais como: modelagem matemática, IA, Pesquisa Operacional, etc
- Ao final voce vai conseguir resolver problemas com alguma complexidade e ler códigos de grandes programadores da área: Barták, Neng-Fa, Hakank, Dymichenko

Apresentação ao Curso de PICAT – V

- Assim, ao final deste curso terás uma visão forte de uma ferramenta computacional, utilizada em várias áreas tais como: modelagem matemática, IA, Pesquisa Operacional, etc
- Ao final voce vai conseguir resolver problemas com alguma complexidade e ler códigos de grandes programadores da área: Barták, Neng-Fa, Hakank, Dymichenko
- Tópicos que serão cobertos no curso:

Histórico

- Criada em 2013 por Neng-Fa Zhou e Jonathan Fruhman
- Utiliza o B-Prolog como base de implementação, tendo a Lógica de Primeira-Ordem (LPO) como parte de seu mecanismo programação

Histórico

- Criada em 2013 por Neng-Fa Zhou e Jonathan Fruhman
- Utiliza o B-Prolog como base de implementação, tendo a Lógica de Primeira-Ordem (LPO) como parte de seu mecanismo programação
- Uma evolução ao Prolog após seus mais de 40 anos de sucesso!

Histórico

- Criada em 2013 por Neng-Fa Zhou e Jonathan Fruhman
- Utiliza o B-Prolog como base de implementação, tendo a Lógica de Primeira-Ordem (LPO) como parte de seu mecanismo programação
- Uma evolução ao Prolog após seus mais de 40 anos de sucesso!
- Sua atual versão é a 2.x (16 de abril de 2019).

Histórico

- Criada em 2013 por Neng-Fa Zhou e Jonathan Fruhman
- Utiliza o B-Prolog como base de implementação, tendo a Lógica de Primeira-Ordem (LPO) como parte de seu mecanismo programação
- Uma evolução ao Prolog após seus mais de 40 anos de sucesso!
- Sua atual versão é a 2.x (16 de abril de 2019).
- Código-aberto, segue as regras da FSF

Conhecendo PICAT

- Picat é uma linguagem de programação simples de usar, poderosa e multi-uso
- Alguma de suas características são associadas com linguagens lógicas, como Prolog, B-Prolog, Goedel, etc

Conhecendo PICAT

- Picat é uma linguagem de programação simples de usar, poderosa e multi-uso
- Alguma de suas características são associadas com linguagens lógicas, como Prolog, B-Prolog, Goedel, etc
- Picat é uma linguagem essencialmente multiparadigma, abrangendo partes de vários paradigmas de programação: declarativo (lógico e funcional) e imperativo

O que é ser Multiparadigma ?

- Paradigma: um conjunto de características baseado em alguma abordagem teórica

O que é ser Multiparadigma ?

- Paradigma: um conjunto de características baseado em alguma abordagem teórica
- Picat é ma linguagem multiparadigma pois abrange os seguintes paradigmas:
  - Lógico
  - Funcional
  - Procedural

O que é ser Multiparadigma ?

- Paradigma: um conjunto de características baseado em alguma abordagem teórica
- Picat é ma linguagem multiparadigma pois abrange os seguintes paradigmas:
  - Lógico
  - Funcional
  - Procedural
- Em resumo, *uma boa mistura* de: Haskell (Funcional) , Prolog (Lógica) e Python (Procedural e Funcional).

Paradigma Lógico

- Uma linguagem lógica é uma onde o programa é expresso como um conjunto de predicados lógicos, escritos por *fatos* e *regras*

Paradigma Lógico

- Uma linguagem lógica é uma onde o programa é expresso como um conjunto de predicados lógicos, escritos por *fatos* e *regras*
- Regras são escritas em formas de cláusulas, as quais são interpretadas como implicações lógicas. Depen dem das premissas serem verdadeiras para esta ser verdadeira.

Paradigma Lógico

- Uma linguagem lógica é uma onde o programa é expresso como um conjunto de predicados lógicos, escritos por *fatos* e *regras*
- Regras são escritas em formas de cláusulas, as quais são interpretadas como implicações lógicas. Depen dem das premissas serem verdadeiras para esta ser verdadeira.
- Fatos são cláusulas sem premissas, verdades absolutas.

Paradigma Lógico

- Uma linguagem lógica é uma onde o programa é expresso como um conjunto de predicados lógicos, escritos por *fatos* e *regras*
- Regras são escritas em formas de cláusulas, as quais são interpretadas como implicações lógicas. Depen dem das premissas serem verdadeiras para esta ser verdadeira.
- Fatos são cláusulas sem premissas, verdades absolutas.
- Este paradigma é a **base** do Picat

Paradigma Funcional

- Uma linguagem funcional é uma onde os elementos do programa podem ser avaliados e tratados como funções matemáticas.

Paradigma Funcional

- Uma linguagem funcional é uma onde os elementos do programa podem ser avaliados e tratados como funções matemáticas.
- Um dos principais motivos em usar linguagens funcionais é a previsibilidade e facilidade no entendimento do estado atual do programa.

Paradigma Funcional

- Uma linguagem funcional é uma onde os elementos do programa podem ser avaliados e tratados como funções matemáticas.
- Um dos principais motivos em usar linguagens funcionais é a previsibilidade e facilidade no entendimento do estado atual do programa.
- Este fato de uma sintaxe simples, torna o Picat intuitivo e legível na funcionalidade de seus códigos.

Paradigma Procedural

- Uma linguagem procedural é uma que pode ser subdividida em *procedimentos*, também chamados de rotinas, subrotinas ou funções

Paradigma Procedural

- Uma linguagem procedural é uma que pode ser subdividida em *procedimentos*, também chamados de rotinas, subrotinas ou funções
- Em linguagens procedurais há um procedimento principal (em geral é chamado de *Main*) que controla o uso e a chamada de outros procedimentos. Em Picat há tal hierarquia.

Paradigma Procedural

- Uma linguagem procedural é uma que pode ser subdividida em *procedimentos*, também chamados de rotinas, subrotinas ou funções
- Em linguagens procedurais há um procedimento principal (em geral é chamado de *Main*) que controla o uso e a chamada de outros procedimentos. Em Picat há tal hierarquia.
- Em Picat, cada premissa é tratada como um procedimento, que é resolvido por meio de métodos de inferência lógica.

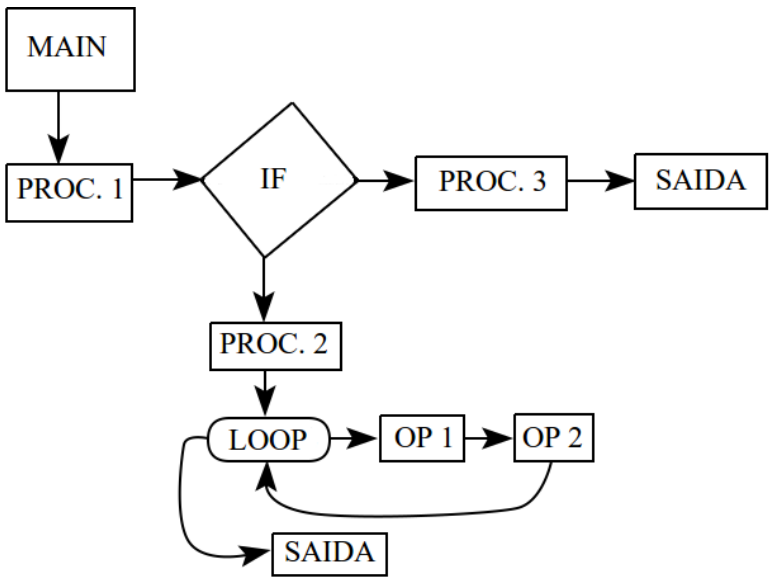


Figura 1: Fluxograma representando a estrutura de um programa Procedural

Algumas Características:

Algumas Características:

- Sintaxe elegante e simples, facilitando a leitura e entendimento do código

Algumas Características:

- Sintaxe elegante e simples, facilitando a leitura e entendimento do código
- Velocidade de execução em um ambiente *interpretado* (há uma *máquina virtual* como Python, Java e alguns Prologs)

Algumas Características:

- Sintaxe elegante e simples, facilitando a leitura e entendimento do código
- Velocidade de execução em um ambiente *interpretado* (há uma *máquina virtual* como Python, Java e alguns Prologs)
- Disponibilidade em vários sistemas operacionais e arquiteturas

Algumas Características:

- Sintaxe elegante e simples, facilitando a leitura e entendimento do código
- Velocidade de execução em um ambiente *interpretado* (há uma *máquina virtual* como Python, Java e alguns Prologs)
- Disponibilidade em vários sistemas operacionais e arquiteturas
- Análogo a Python, podem ser feitas *queries* ou *consultas* ao terminal de Picat.

Algumas Características:

- Sintaxe elegante e simples, facilitando a leitura e entendimento do código
- Velocidade de execução em um ambiente *interpretado* (há uma *máquina virtual* como Python, Java e alguns Prologs)
- Disponibilidade em vários sistemas operacionais e arquiteturas
- Análogo a Python, podem ser feitas *queries* ou *consultas* ao terminal de Picat.
- Há várias bibliotecas da própria linguagem, e diversas ferramentas externas permitindo o incremento do poder do Picat.

Acrônimo de **P.I.C.A.T.** I

P: *Pattern-matching*: Utiliza o conceito de *casamento de padrões* entre objetos, bem como os conceitos da *unificação* da LPO

I: *Intuitive*: Oferece estruturas de decisão, atribuição e laços de repetição, etc. Análogo a outras linguagens de programação mais populares

C: *Constraints*: Suporta a programação por restrições (PR) para problemas combinatórios

A: *Actors*: Suporte as chamadas a eventos, os atores;

T: *Tabling*: Implementa a técnica de *memoization*, com soluções imediatas para problemas de Programação Dinâmica (PD).

Instalação do PICAT

- Baixar a versão desejada de:

  `http://picat-lang.org/download.html`

Instalação do PICAT

- Baixar a versão desejada de:

  `http://picat-lang.org/download.html`

- Descompactar. Em geral em: **/usr/local/Picat/** no Linux e IOS

Instalação do PICAT

- Baixar a versão desejada de:

  `http://picat-lang.org/download.html`

- Descompactar. Em geral em: **/usr/local/Picat/** no Linux e IOS
- Criar um link simbólico (Linux) ou atalhos (Windows):

  `ln -s /usr/local/Picat/picat  /usr/bin/picat`

Instalação do PICAT

- Baixar a versão desejada de:

  `http://picat-lang.org/download.html`

- Descompactar. Em geral em: **/usr/local/Picat/** no Linux e IOS
- Criar um link simbólico (Linux) ou atalhos (Windows):

  `ln -s /usr/local/Picat/picat /usr/bin/picat`

- Se quiser adicionar (opcional) uma variável de ambiente:

  ```
  PICATPATH=/usr/local/Picat/
  export PICATPATH
  ```

Instalação do PICAT

- Baixar a versão desejada de:

  `http://picat-lang.org/download.html`

- Descompactar. Em geral em: **/usr/local/Picat/** no Linux e IOS
- Criar um link simbólico (Linux) ou atalhos (Windows):

  `ln -s /usr/local/Picat/picat /usr/bin/picat`

- Se quiser adicionar (opcional) uma variável de ambiente:

  ```
  PICATPATH=/usr/local/Picat/
  export PICATPATH
  ```

- Ou ainda, adicione o caminho:

  `PATH=$PATH:/usr/local/Picat`

Instalação do PICAT

- Baixar a versão desejada de:

  `http://picat-lang.org/download.html`

- Descompactar. Em geral em: **/usr/local/Picat/** no Linux e IOS
- Criar um link simbólico (Linux) ou atalhos (Windows):

  `ln -s /usr/local/Picat/picat /usr/bin/picat`

- Se quiser adicionar (opcional) uma variável de ambiente:

  ```
  PICATPATH=/usr/local/Picat/
  export PICATPATH
  ```

- Ou ainda, adicione o caminho:

  `PATH=$PATH:/usr/local/Picat`

- Finalmente, tenha um editor de texto apropriado.

  Sugestão: *Geany*, *Sublime* ou *Atom*.

Instalação do PICAT

- Baixar a versão desejada de:
  `http://picat-lang.org/download.html`
- Descompactar. Em geral em: **/usr/local/Picat/** no Linux e IOS
- Criar um link simbólico (Linux) ou atalhos (Windows):
  `ln -s /usr/local/Picat/picat /usr/bin/picat`
- Se quiser adicionar (opcional) uma variável de ambiente:
  ```
  PICATPATH=/usr/local/Picat/
  export PICATPATH
  ```
- Ou ainda, adicione o caminho:
  `PATH=$PATH:/usr/local/Picat`
- Finalmente, tenha um editor de texto apropriado.
  Sugestão: *Geany*, *Sublime* ou *Atom*.
- Se não tiver *plugin* para Picat, escolha a sintaxe da linguagem *Erlang*.

Usando Picat

- Picat é uma linguagem disponível em qualquer arquitetura de processamento.

Usando Picat

- Picat é uma linguagem disponível em qualquer arquitetura de processamento.
- Qualquer emergência, o ambiente completo de execução do Picat pode ser reconstruído a partir da linguagem C padrão

Usando Picat

- Picat é uma linguagem disponível em qualquer arquitetura de processamento.
- Qualquer emergência, o ambiente completo de execução do Picat pode ser reconstruído a partir da linguagem C padrão
- Os seus arquivos fontes utilizam a extensão **.pi**. Exemplo: `programa.pi`
- Há dois modos principais de utilização do Picat:
  - Modo interativo, onde seu código é digitado e compilado diretamente na linha de comando;
  - *Modo console* onde o console só é utilizado para compilar seus programas.

Usando Picat

- Picat é uma linguagem disponível em qualquer arquitetura de processamento.
- Qualquer emergência, o ambiente completo de execução do Picat pode ser reconstruído a partir da linguagem C padrão
- Os seus arquivos fontes utilizam a extensão **.pi**. Exemplo: `programa.pi`
- Há dois modos principais de utilização do Picat:
  - Modo interativo, onde seu código é digitado e compilado diretamente na linha de comando;
  - *Modo console* onde o console só é utilizado para compilar seus programas.
- Códigos executáveis 100% **stand-alone**: ainda não!
- Neste quesito, estamos em igualdade com Java, Prolog e Python

Exemplo – Conceitos

Acompanhar as explicações do código de:
`https://github.com/claudiosa/CCS/blob/master/picat/alo_mundo.pi`

```
main => msg_01  ,
        msg_02 .

msg_01 =>  printf("  ALO MUNDO!!! ").
msg_02 =>  printf("\n  FIM \n").
```

Execução na Console Linux ou Windows

```
$ picat alo_mundo.pi
  ALO MUNDO!!!
  FIM
$
```

Execução no Ambiente do Interpretador

```
$ picat
Picat 2.0, (C) picat-lang.org, 2013-2016.
Type 'help' for help.
Picat> cl(álo_mundo.pi').
Compiling:: alo_mundo.pi
alo_mundo.pi compiled in 0 milliseconds
loading...

yes

Picat> main
  ALO MUNDO!!!
  FIM

yes

Picat> msg_02

  FIM
```

Reflexões

- O conteúdo desta parte do curso pode ser complementado com a **Videoaula 01: Introdução ao PICAT**, disponível no Youtube:
  `https://www.youtube.com/watch?v=0DmTyFFQPK8`

Reflexões

- O conteúdo desta parte do curso pode ser complementado com a **Videoaula 01: Introdução ao PICAT**, disponível no Youtube:
  `https://www.youtube.com/watch?v=0DmTyFFQPK8`
- Para próxima seção esteja com o Picat instalado em seu computador para um melhor aproveitamento.

Programação por Restrições (PR) – I

- A Programação por Restrições (PR) é conhecida por *Constraint Programming* ou simplesmente **CP**

Programação por Restrições (PR) – I

- A Programação por Restrições (PR) é conhecida por *Constraint Programming* ou simplesmente **CP**
- Uma poderosa teoria (e técnica) que contorna a complexidade de certos problemas exponenciais

Programação por Restrições (PR) – I

- A Programação por Restrições (PR) é conhecida por *Constraint Programming* ou simplesmente **CP**
- Uma poderosa teoria (e técnica) que contorna a complexidade de certos problemas exponenciais
- A PR encontrava-se inicialmente dentro da IA e PO, mas como várias outras, tornaram-se fortes e autônomas. Atualmente uma área de pesquisa bem forte em alguns países.

Programação por Restrições (PR) – II

- Aproximadamente o algoritmo da PR é dado:

Programação por Restrições (PR) – II

- Aproximadamente o algoritmo da PR é dado:
  1. Avaliar algebricamente os domínios das variáveis com suas restrições
  2. Intercala iterativamente a propagação de restrições com um algoritmo de busca
  3. A cada variável instanciada, o processo é repetido sobre as demais variáveis, reduzindo progressivamente o espaço de busca
  4. Volte ao passo inicial até que os domínios permaneçam estáticos e que as variáveis apresentem instâncias consistentes

Programação por Restrições (PR) – II

- Aproximadamente o algoritmo da PR é dado:
  1. Avaliar algebricamente os domínios das variáveis com suas restrições
  2. Intercala iterativamente a propagação de restrições com um algoritmo de busca
  3. A cada variável instanciada, o processo é repetido sobre as demais variáveis, reduzindo progressivamente o espaço de busca
  4. Volte ao passo inicial até que os domínios permaneçam estáticos e que as variáveis apresentem instâncias consistentes
- Este núcleo é uma busca por constantes otimizações

Programação por Restrições (PR) – II

- Aproximadamente o algoritmo da PR é dado:
  1. Avaliar algebricamente os domínios das variáveis com suas restrições
  2. Intercala iterativamente a propagação de restrições com um algoritmo de busca
  3. A cada variável instanciada, o processo é repetido sobre as demais variáveis, reduzindo progressivamente o espaço de busca
  4. Volte ao passo inicial até que os domínios permaneçam estáticos e que as variáveis apresentem instâncias consistentes
- Este núcleo é uma busca por constantes otimizações
- Uma das virtudes da PR: a legibilidade e clareza de suas soluções

Programação por Restrições (PR) – III

- Problemas combinatoriais com domínio nos inteiros são bons candidatos a serem resolvidos por PR

Programação por Restrições (PR) – III

- Problemas combinatoriais com domínio nos inteiros são bons candidatos a serem resolvidos por PR
- Quando temos problemas que precisamos conhecer **todas** as respostas, não apenas a melhor resposta

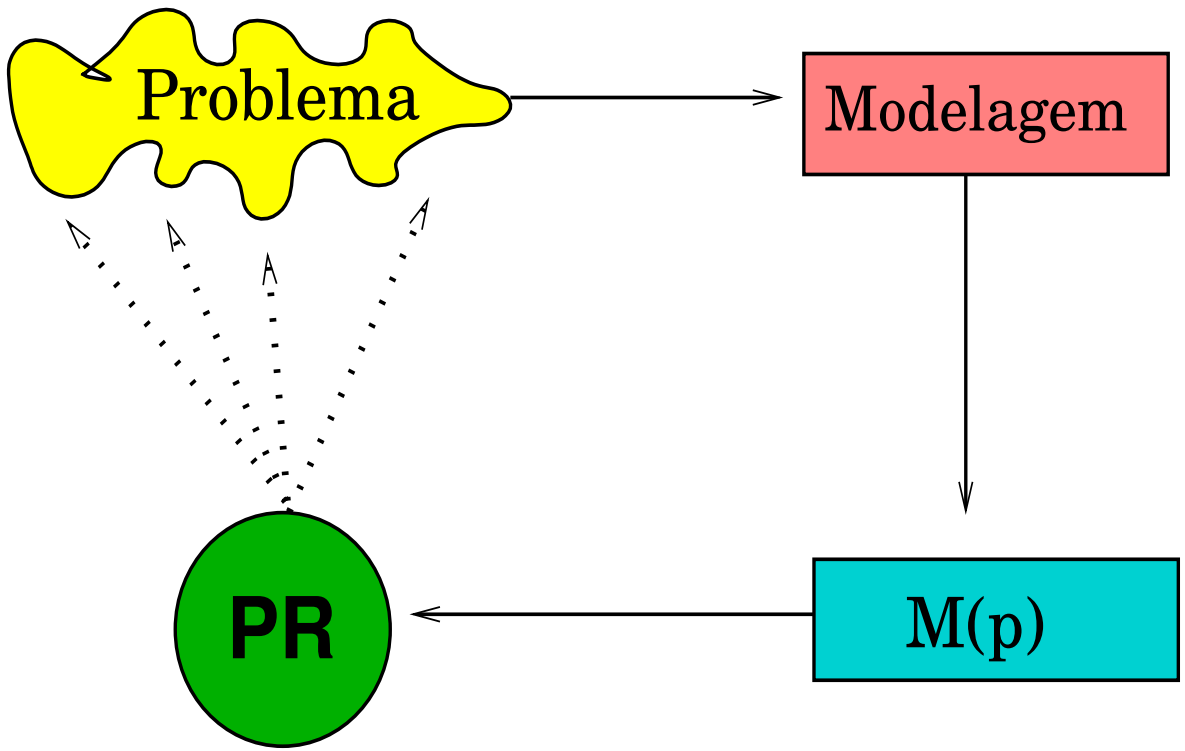
Programação por Restrições (PR) – III

- Problemas combinatoriais com domínio nos inteiros são bons candidatos a serem resolvidos por PR
- Quando temos problemas que precisamos conhecer **todas** as respostas, não apenas a melhor resposta
- Quando necessitamos de respostas *precisas* e não apenas as aproximadas. Há um custo computacional a ser pago aqui!

Programação por Restrições (PR) – III

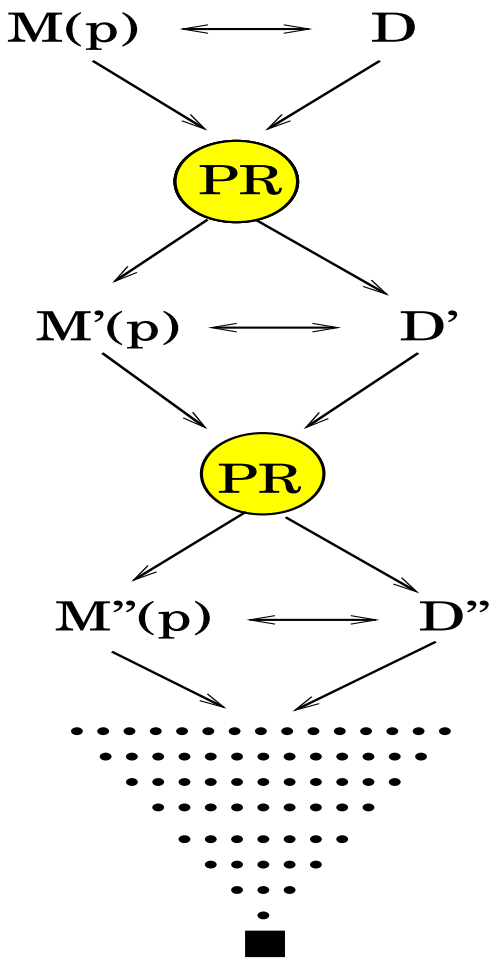
- Problemas combinatoriais com domínio nos inteiros são bons candidatos a serem resolvidos por PR
- Quando temos problemas que precisamos conhecer **todas** as respostas, não apenas a melhor resposta
- Quando necessitamos de respostas *precisas* e não apenas as aproximadas. Há um custo computacional a ser pago aqui!
-

Metodologia da Construção de Modelos



Um modelo

Fluxo de Cálculo da PR



Onde o objetivo da PR é:

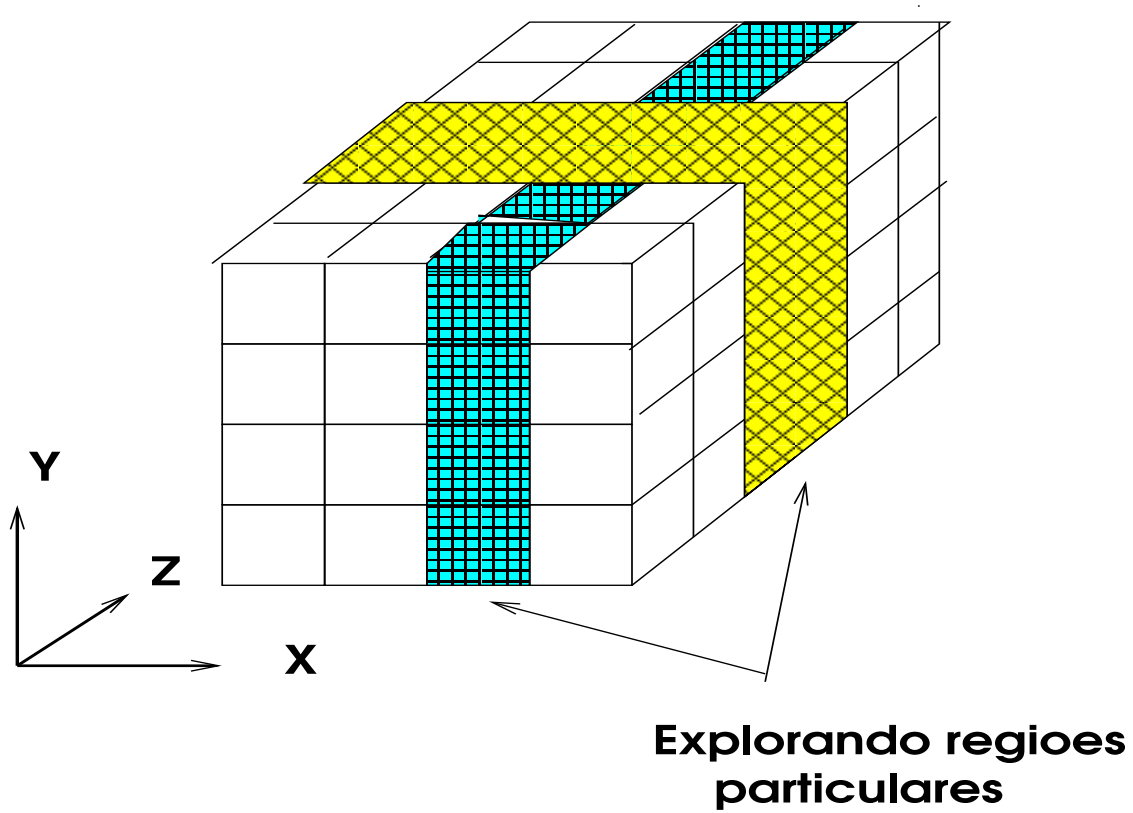


Figura 2: Realizar buscas com regiões reduzidas – promissoras (regiões factíveis de soluções)

Redução Iterativa em Sub-problemas

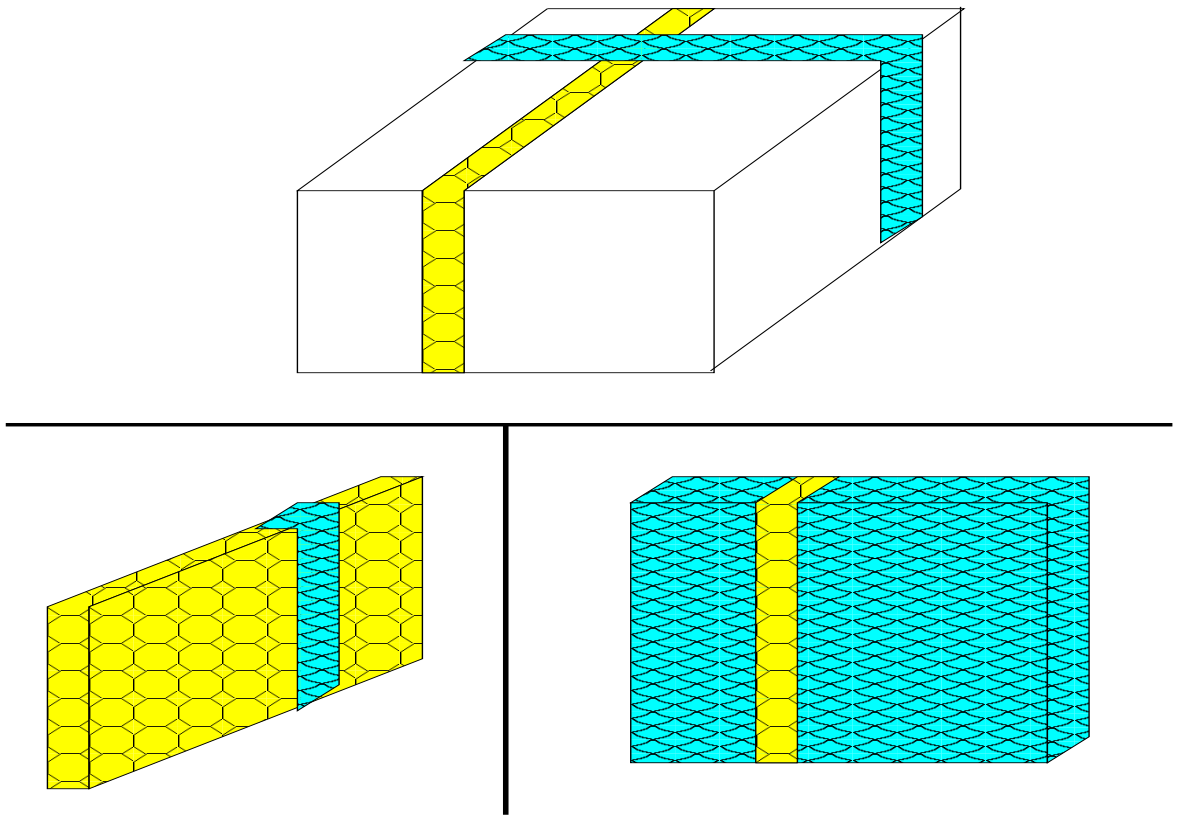


Figura 3: Redução de um CP em outros sub-problemas CPs equivalentes

Exemplo

Dado um n'umero, encontre um par de numeros primos, diferentes entre si, que somados deêm este n'umero par.
Exemplo:
Seja o PAR = 18
Uma soluç ao:
$N_1 = 7$ e $N_2 = 11$
pois
$N_1 + N_2 = 18$

Código Completo

- Acompanhar as explicações do código de: `https://github.com/claudiosa/CCS/blob/master/picat/soma_N1_N2_primos_CP.pi`
- Confira a execução e testes

Código em Partes

```
modelo =>
    PAR = 382,
    Variaveis = [N1,N2],
    % Gerando um domino soh de primos
    % L_dom = [I : I in 1..1000, eh_primo(I) == true],   %OU
    L_dom = [I : I in 1..1000, prime(I)],
  Variaveis :: L_dom,
```

Código em Partes

```
    % RESTRICOES
    N1 #!= N2,
    N1 #< PAR,
    N2 #< PAR,
    N1 + N2 #= PAR,

% A BUSCA
solve([ff], Variaveis),
  % UMA SAIDA
printf("\n  N1: %d\t N2: %d", N1,N2),
printf("\n.....................................")
.
```

Código em Partes

```
import cp.

% main => modelo .
% main ?=> modelo, fail.
% main =>  true.

main =>
    L = findall(_, $modelo),
    writef("\n Total de solucoes:  %d \n", length(L)) .
```

Saída

```
Picat> cl('soma_N1_N2_primos_CP').
Compiling:: soma_N1_N2_primos_CP.pi
** Warning  : redefine_preimported_symbol(math): prime / 1
soma_N1_N2_primos_CP.pi compiled in 7 milliseconds
loading...

yes

Picat> main.

  N1: 3  N2: 379
.....................................
  N1: 23  N2: 359
.....................................
  N1: 29  N2: 353
.....................................

 .....................................
```

Reflexões

- Há outros métodos para se resolver estes problemas. Exemplo: Programação Linear, Buscas Heurísticas

Reflexões

- Há outros métodos para se resolver estes problemas.
  Exemplo: Programação Linear, Buscas Heurísticas
- A área é extensa, contudo, Picat adere há todos requisitos da PR