

Escalonando disciplinas por Programação Lógica

Grupo da disciplina Laboratório de Métodos Ágeis

Apresentadores: Ana, Briza, Daniel, Lorenzo, Jessica e Bruno

Contexto e motivação



Júpiter - Sistema de Gestão Acadêmica da Pró-Reitoria de Graduação

Instituto de Matemática e Estatística

Ciência da Computação

Disciplina: MAC0472 - Laboratório de Métodos Ágeis
Agile Methods Lab

Créditos Aula: 4

Créditos Trabalho: 2

Carga Horária Total: 120 h

Tipo: Semestral

Ativação: 01/01/2018 **Desativação:**

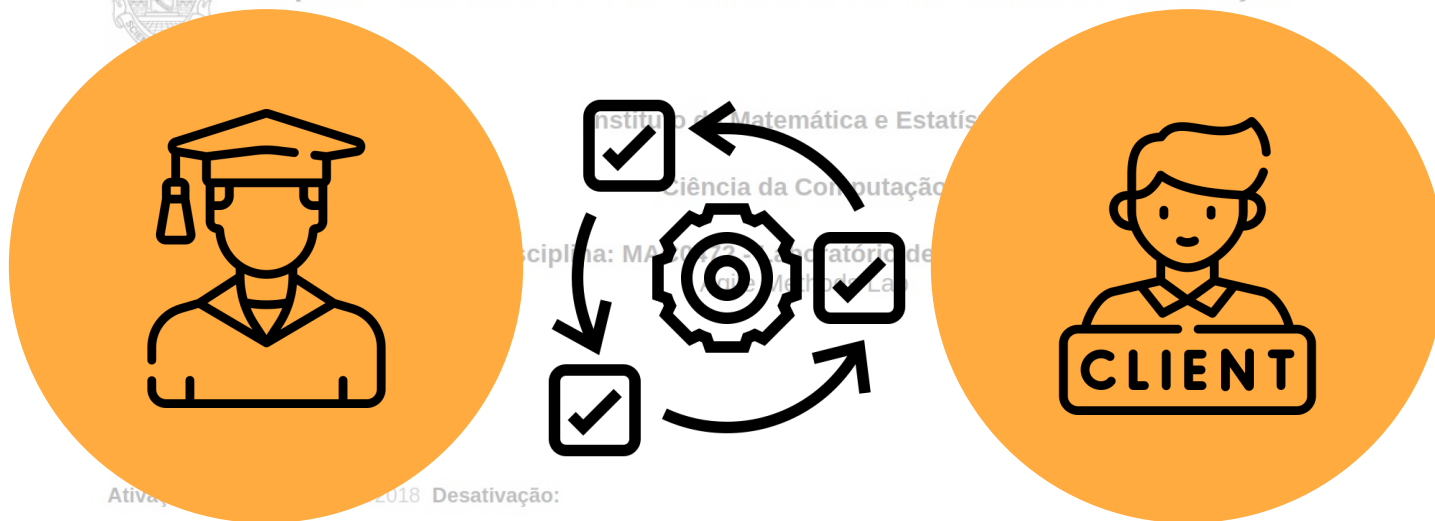
Objetivos

Familiarizar o estudante com metodologias ágeis de desenvolvimento de software orientado a objetos.

Contexto e motivação



Júpter - Sistema de Gestão Acadêmica da Pró-Reitoria de Graduação



Objetivos

Familiarizar o estudante com metodologias ágeis de desenvolvimento de software orientado a objetos.

Contexto e motivação



- Escalonamento automático de disciplinas MAC
- Considerar todas as restrições necessárias da grade do DCC
- Desenvolvimento em ASP

Grade Horária

HORÁRIOS 1º SEMESTRE 2023							Legenda
	SEGUNDA	TERÇA	QUARTA	QUINTA	SEXTA		
8h - 10h	MAC0110 MAE0228 Aneis&Corpos Paralela TopicosGrafos	Booleana LabNum ML	Estocasticos AnaliseReal SO CG	ED2 LabJef IA OtInteira	Fumac LabBD		1º Ano
10h - 12h	SO AnaliseReal OtInteira	ED2 LabJef LabBD IA	MAC0110 MAE0228 Aneis&Corpos Paralela	Booleana ML LabNum TopicosGrafos	Estocasticos DesafiosAvanc AnaliseReal CG		2º Ano
13:00 - 14:40		Palestrinha 1		Palestrinha 2			3º Ano
14h - 16h	Modelagem TecProg2 Empreendedorismo	Grafinhos	Fumac DesafiosAvanc	Visao OtNaoLin Cripto TeoNumeros			Sistemas, IA
16h - 18h	Modelagem Empreendedorismo OtNaoLin MatDiscreta	Visao Cripto TeoNumeros	TecProg2 MatDiscreta	Grafinhos			Teoria, E-Science, Outros

+OtNaoLin (seg-19:20, qui-21:10)

Grade Horária

Como é feita?

COMPADI

decide as disciplinas oferecidas e os professores que as ministrarão no semestre



Comissão
de Horário

monta a grade horária

Grade Horária

Não é tão simples..



Grade Horária



Reuniões longas e
trabalhosas



Todo começo de
semestre

Grade Horária

Automatização

- grande volume de dados e restrições
- escalonamento semestral
- manualmente desgastante e demorado
- problema NP-difícil → satisfatibilidade

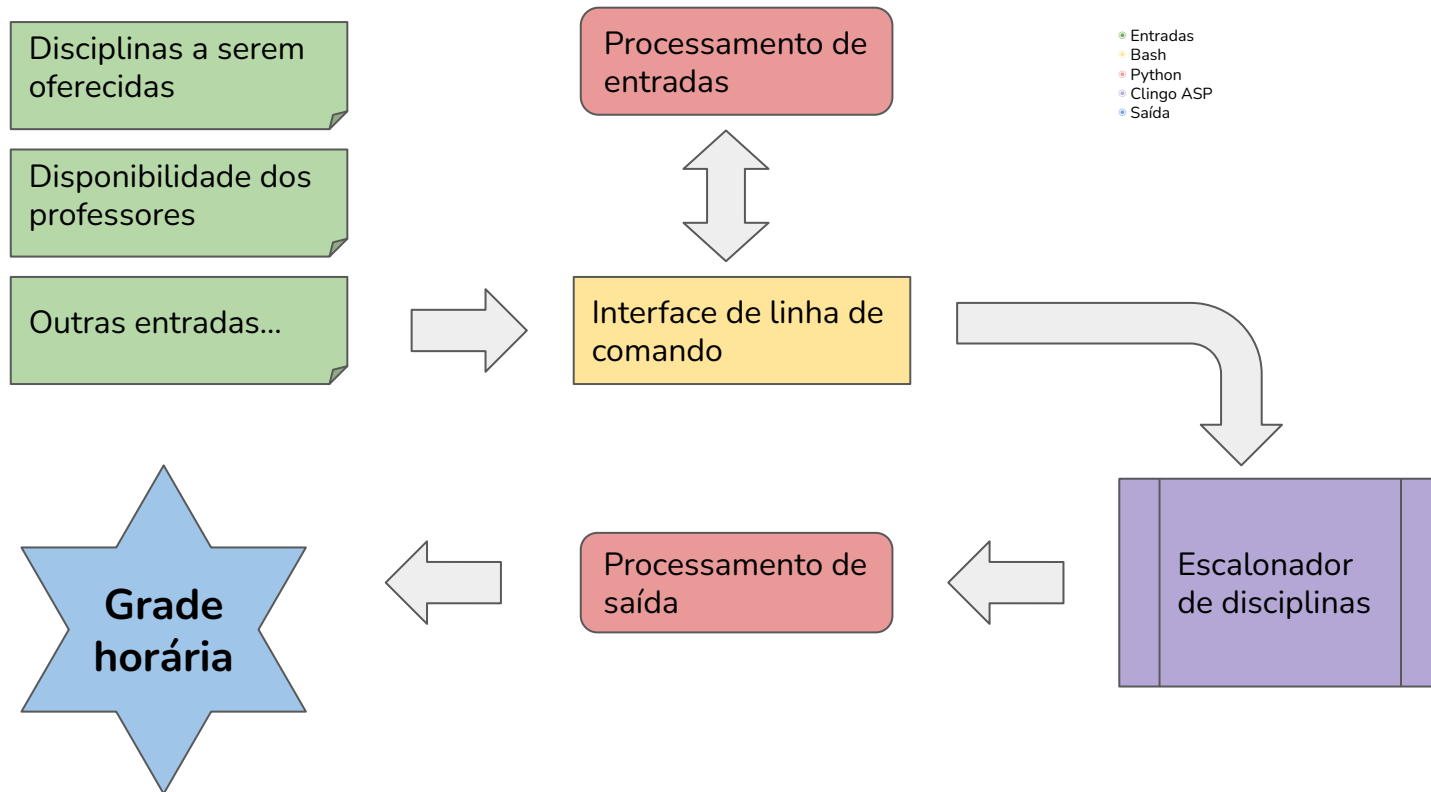
Ferramentas utilizadas

Problema complexo, várias ferramentas necessárias:

- Clingo ASP → escalonador de disciplinas
- Python → processamento de entradas e saídas
- Bash → interface de linha de comando
- Docker, Git, Excel...



Fluxo da aplicação



Por que ASP?

Como todo problema de computação, muitas soluções possíveis:

Por que ASP?

Como todo problema de computação, muitas soluções possíveis:

- Programação procedural

Por que ASP?

Como todo problema de computação, muitas soluções possíveis:

- Programação procedural → além de descrever as regras do problema precisamos implementar um mecanismo de otimização eficiente e correto (bem difícil)

Por que ASP?

Como todo problema de computação, muitas soluções possíveis:

- Programação procedural → além de descrever as regras do problema precisamos implementar um mecanismo de otimização eficiente e correto (bem difícil)
- Modelo de aprendizado de máquina

Por que ASP?

Como todo problema de computação, muitas soluções possíveis

- Programação procedural → além de descrever as regras do problema precisamos implementar um mecanismo de otimização eficiente e correto (bem difícil)
- Modelo de aprendizado de máquina → sem *dataset* prévio, necessidade de *fine tuning* e treinamento toda vez que uma nova disciplina é adicionada à grade

ASP como solução

“Answer Set Programming (ASP) offers a simple and powerful modeling language to solve combinatorial problems.

With our tools you can concentrate on an actual problem, rather than a smart way of implementing it.”

- potassco.org



O que isso significa?

Se o nosso problema for um problema combinatório, podemos nos preocupar apenas em modelar o problema, já que a busca por uma solução e a otimização ficam por conta do ASP!



Problemas combinatórios

Um problema combinatório é composto por:

Problemas combinatórios

Um problema combinatório é composto por:

- Um conjunto **finito** de possíveis soluções

Problemas combinatórios

Um problema combinatório é composto por:

- Um conjunto **finito** de possíveis soluções
- Um conjunto bem definido de **restrições** sobre as potenciais soluções do problema

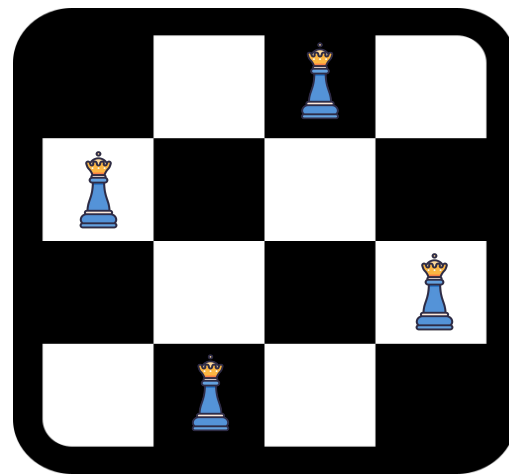
Problemas combinatórios

Um problema combinatório é composto por:

- Um conjunto **finito** de possíveis soluções
- Um conjunto bem definido de **restrições** sobre as potenciais soluções do problema
- (OPCIONAL) Um critério de **otimização** sobre a solução encontrada

Exemplo: N-Rainhas

Problema: posicionar N rainhas em um tabuleiro $N \times N$ de forma que nenhuma das rainhas possa atacar outra rainha. Uma rainha pode atacar a outra se ambas estiverem na mesma linha, coluna ou diagonal.




Solução para o problema das N rainhas com $N = 4$


N-Rainhas é combinatório?

- Um conjunto **finito** de possíveis soluções



N-Rainhas é combinatório?

- Um conjunto **finito** de possíveis soluções 
 - Todas as possíveis formas de posicionar N rainhas em um tabuleiro NxN



N-Rainhas é combinatório?

- Um conjunto **finito** de possíveis soluções 
 - Todas as possíveis formas de posicionar N rainhas em um tabuleiro NxN
- Um conjunto bem definido de **restrições** sobre as potenciais soluções do problema




N-Rainhas é combinatório?

- Um conjunto **finito** de possíveis soluções 
 - Todas as possíveis formas de posicionar N rainhas em um tabuleiro NxN
- Um conjunto bem definido de **restrições** sobre as potenciais soluções do problema 
 - Todas as rainhas devem ser posicionadas no tabuleiro
 - Nenhum par de rainhas pode estar na mesma linha, coluna ou diagonal

N-Rainhas é combinatório?

- Um conjunto **finito** de possíveis soluções 
 - Todas as possíveis formas de posicionar N rainhas em um tabuleiro NxN
- Um conjunto bem definido de **restrições** sobre as potenciais soluções do problema 
 - Todas as rainhas devem ser posicionadas no tabuleiro
 - Nenhum par de rainhas pode estar na mesma linha, coluna ou diagonal
- (OPCIONAL) Um critério de **otimização** sobre a solução encontrada

N-Rainhas é combinatório?

- Um conjunto **finito** de possíveis soluções 
 - Todas as possíveis formas de posicionar N rainhas em um tabuleiro NxN
- Um conjunto bem definido de **restrições** sobre as potenciais soluções do problema 
 - Todas as rainhas devem ser posicionadas no tabuleiro
 - Nenhum par de rainhas pode estar na mesma linha, coluna ou diagonal
- (OPCIONAL) Um critério de **otimização** sobre a solução encontrada 
 - Não se aplica!

E o escalonamento...?


Será que conseguimos enquadrar nosso problema de escalonamento de disciplinas em um problema combinatório?




Escalonamento: um problema combinatório!

- Um conjunto **finito** de possíveis soluções



Escalonamento: um problema combinatório!

- Um conjunto **finito** de possíveis soluções 
 - É o conjunto de todas as formas possíveis de organizar as disciplinas que devem ser lecionadas nos horários de aula disponíveis.
 - É um número **enorme** de possibilidades (permutações de permutações de permutações...), mas é finito.



Escalonamento: um problema combinatório!

- Um conjunto **finito** de possíveis soluções 
 - É o conjunto de todas as formas possíveis de organizar as disciplinas que devem ser lecionadas nos horários de aula disponíveis.
 - É um número **enorme** de possibilidades (permutações de permutações de permutações...), mas é finito.
- Um conjunto bem definido de **restrições** sobre as potenciais soluções do problema




Escalonamento: um problema combinatório!

- Um conjunto **finito** de possíveis soluções 
 - É o conjunto de todas as formas possíveis de organizar as disciplinas que devem ser lecionadas nos horários de aula disponíveis.
 - É um número **enorme** de possibilidades (permutações de permutações de permutações...), mas é finito.
- Um conjunto bem definido de **restrições** sobre as potenciais soluções do problema 
 - Todas as disciplinas que serão oferecidas devem ser posicionadas na grade horária
 - O número de aulas semanais de cada disciplina deve ser respeitado
 - Duas disciplinas oferecidas pelo mesmo professor não podem ser dadas no mesmo horário
 - ...

Escalonamento: um problema combinatório!

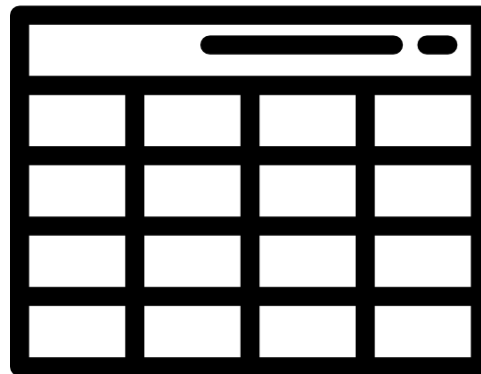
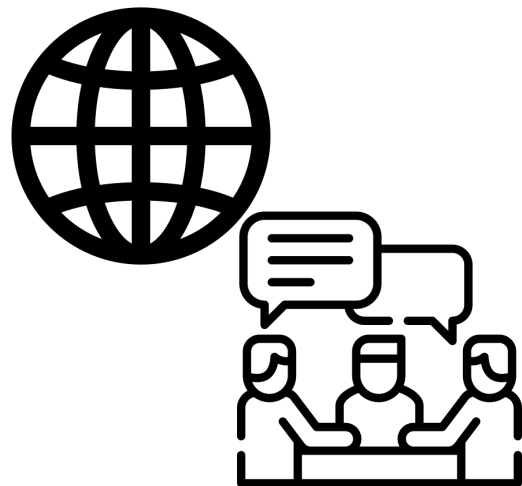
- Um conjunto **finito** de possíveis soluções 
 - É o conjunto de todas as formas possíveis de organizar as disciplinas que devem ser lecionadas nos horários de aula disponíveis.
 - É um número **enorme** de possibilidades (permutações de permutações de permutações...), mas é finito.
- Um conjunto bem definido de **restrições** sobre as potenciais soluções do problema 
 - Todas as disciplinas que serão oferecidas devem ser posicionadas na grade horária
 - O número de aulas semanais de cada disciplina deve ser respeitado
 - Duas disciplinas oferecidas pelo mesmo professor não podem ser dadas no mesmo horário
 - ...
- (OPCIONAL) Um critério de **otimização** sobre a solução encontrada

Escalonamento: um problema combinatório!

- Um conjunto **finito** de possíveis soluções 
 - É o conjunto de todas as formas possíveis de organizar as disciplinas que devem ser lecionadas nos horários de aula disponíveis.
 - É um número **enorme** de possibilidades (permutações de permutações de permutações...), mas é finito.
- Um conjunto bem definido de **restrições** sobre as potenciais soluções do problema 
 - Todas as disciplinas que serão oferecidas devem ser posicionadas na grade horária
 - O número de aulas semanais de cada disciplina deve ser respeitado
 - Duas disciplinas oferecidas pelo mesmo professor não podem ser dadas no mesmo horário
 - ...
- (OPCIONAL) Um critério de **otimização** sobre a solução encontrada 
 - Um pouco mais complicado... vamos entender mais o problema primeiro!

Formalização da Ideia

Inputs



Formalização da Ideia

Inputs

Grade Curricular

Legenda: CH=Carga horária Total; CE=Carga horária de Estágio; CP=Carga horária de Práticas como Componentes Curriculares;
ATPA=Carga horária em Atividades Teórico-Práticas de Aprofundamento

Disciplinas Obrigatórias		Créd. Aula	Créd. Trab.	CH	CE	CP	ATPA
1º Período Ideal							
<u>MAC0101</u>	Integração na Universidade e na Profissão		2	0	30		
<u>MAC0105</u>	Fundamentos de Matemática para a Computação		4	0	60		
<u>MAC0110</u>	Introdução à Computação		4	0	60		
<u>MAC0329</u>	Álgebra Booleana e Aplicações no Projeto de Arquitetura de Computadores		4	0	60		
<u>MAT0112</u>	Vetores e Geometria		4	0	60		
<u>MAT2453</u>	Cálculo Diferencial e Integral I		6	0	90		
Subtotal:			24	0	360		
2º Período Ideal							
<u>MAC0121</u>	Algoritmos e Estruturas de Dados I		4	0	60		
<u>MAC0110 - Introdução à Computação</u>				Requisito			
<u>MAC0216</u>	Técnicas de Programação I		4	2	120		
<u>MAC0110 - Introdução à Computação</u>				Requisito			
<u>MAC0239</u>	Introdução à Lógica e Verificação de Programas		4	0	60		
<u>MAC0110 - Introdução à Computação</u>				Requisito			
<u>MAE0119</u>	Introdução à Probabilidade e à Estatística		6	0	90	0	
<u>MAT0122</u>	Álgebra Linear I		4	0	60		
<u>MAT0112 - Vetores e Geometria</u>				Requisito			
<u>MAT2454</u>	Cálculo Diferencial e Integral II		4	0	60		
<u>MAT2453 - Cálculo Diferencial e Integral I</u>				Requisito			
Subtotal:			26	2	450		
3º Período Ideal							
<u>MAC0102</u>	Caminhos no Bacharelado em Ciência da Computação		2	0	30		
<u>MAC0121 - Algoritmos e Estruturas de Dados I</u>				Requisito			
<u>MAC0209</u>	Modelagem e Simulação		4	0	60		
<u>MAC0110 - Introdução à Computação</u>				Requisito			
<u>MAC0210</u>	Laboratório de Métodos Numéricos		4	0	60		
<u>MAC0110 - Introdução à Computação</u>				Requisito			
<u>MAT0122 - Álgebra Linear I</u>				Requisito			
<u>MAC0323</u>	Algoritmos e Estruturas de Dados II		4	2	120		
<u>MAC0121 - Algoritmos e Estruturas de Dados I</u>				Requisito			

Formalização da Ideia

Inputs

[INSTITUTO](#)[DEPARTAMENTOS](#)[GRADUAÇÃO](#)[PÓS-GRADUAÇÃO](#)[PESQUISA](#)[EXTENSÃO](#)[INTERNACIONAL](#)[PROJETOS DE PESQUISA](#)[CENTROS](#)[SP JOURNAL](#)[BIBLIOTECA](#)[IME JÚNIOR](#)[Apresentação](#)[Coordenação](#)[Alunos ativos](#)[Áreas de pesquisa](#)[Calendários](#)[Disciplinas](#)[Formulários](#)[Ingresso](#)[Normas](#)[Orientadores](#)

Relação de disciplinas – Pós-graduação em Ciência da Computação

Sigla	Nome da disciplina
GEN5711	Preparação à Docência de Graduação
IBI5037	Algoritmos em Bioinformática
IME4002	Redação Científica em Inglês com Foco na Publicação Internacional: do Texto ao Contexto
MAC4722	Linguagens, Autômatos e Computabilidade
MAC5700	Seminários em Ciência da Computação

Formalização da Ideia

Inputs

- Trilhas da graduação
- Área da Pós-graduação
- Matérias com sigla dupla
- Matérias mais requisitadas
- Matérias com aula dupla

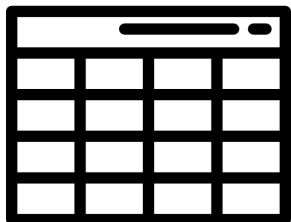
Formalização da Ideia

Inputs

- Informações semestrais
- Padronização junto da COMPADI
 - Nomes e horários de professores
 - Siglas das matérias
 - Turmas, Discip. Serviço, Horários Fixos

Formalização da Ideia

Representação dos dados coletados no programa



```
curriculum(mac0219,data,0).  
curriculum(mac0431,data,0).  
curriculum(mac0425,ia,1).
```

```
num_classes(mac0101,1).  
num_classes(mac0105,2).  
num_classes(mac0110,2).
```

```
obligatory(mac0102,3).  
obligatory(mac0209,3).  
obligatory(mac0110,1).  
obligatory(mac0329,1).
```

```
joint(mac0450,mac5727).  
joint(mac0472,mac5716).
```

```
double(mac0318).  
double(mac0209).
```

```
requiredElective(statistics,mae0532).  
requiredElective(sciences,4302112).
```

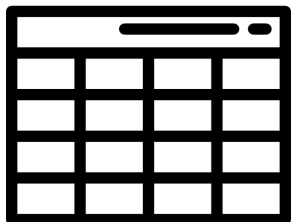
```
highDemand(mac5711).  
highDemand(mac5716).  
highDemand(mac5717).
```

```
course(mac0110,1,gold).  
course(mac0350,1,jef).  
course(mac0499,1,nina).
```

```
available(gold,422).  
available(gold,122).  
available(renata,211).
```

Formalização da Ideia

Representação dos dados coletados no programa



```
curriculum(mac0219,data,0).  
curriculum(mac0431,data,0).  
curriculum(mac0425,ia,1).
```

```
num_classes(mac0101,1).  
num_classes(mac0105,2).  
num_classes(mac0110,2).
```

```
obligatory(mac0102,3).  
obligatory(mac0209,3).  
obligatory(mac0110,1).  
obligatory(mac0329,1).
```

```
joint(mac0450,mac5727).  
joint(mac0472,mac5716).
```

```
double(mac0318).  
double(mac0209).
```

```
requiredElective(statistics,mae0532).  
requiredElective(sciences,4302112).
```

```
highDemand(mac5711).  
highDemand(mac5716).  
highDemand(mac5717).
```

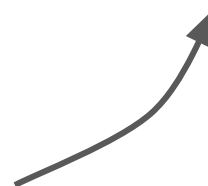
```
course(mac0110,1,gold).  
course(mac0350,1,jef).  
course(mac0499,1,nina).
```

```
available(gold,422).  
available(gold,122).  
available(renata,211).
```

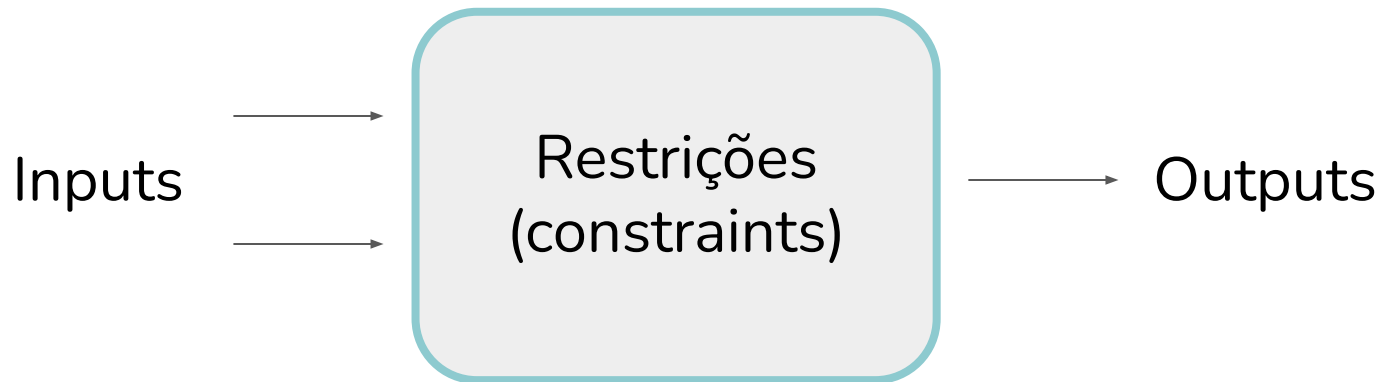
Qual é a saída esperada?

HORÁRIOS 1º SEMESTRE 2023					
	SEGUNDA	TERÇA	QUARTA	QUINTA	SEXTA
8h - 10h	MAC0110 MAE0228 Aneis&Corpos Paralela TopicosGrafos	Booleana LabNum ML	Estocasticos AnaliseReal SO CG	ED2 LabJef IA Otlnteira	Fumac LabBD
10h - 12h	SO AnaliseReal Otlnteira	ED2 LabJef LabBD IA	MAC0110 MAE0228 Aneis&Corpos Paralela	Booleana ML LabNum TopicosGrafos	Estocasticos DesafiosAvanc AnaliseReal CG
13:00 - 14:40		Palestrinha 1		Palestrinha 2	
14h - 16h	Modelagem TecProg2 Empreendedorismo	Grafinhos	Fumac DesafiosAvanc	Visao OtnaoLin Cripto TeoNumeros	
16h - 18h	Modelagem Empreendedorismo OtnaoLin MatDiscreta	Visao Cripto TeoNumeros	TecProg2 MatDiscreta	Grafinhos	
18:30 - 20:10					
19:20 - 21h	OtnaoLin				
21:10 - 22:50				OtnaoLin	

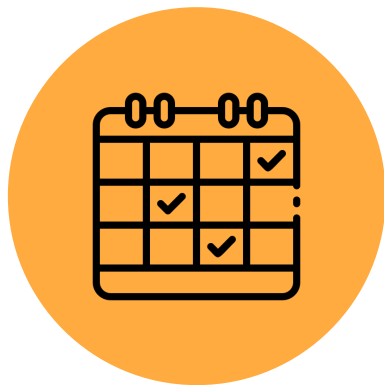
Uma grade que simule bem o resultado do trabalho manual da Comissão



Por que restrições?



Regras de mundo



Todas as classes
devem ser dadas,
N vezes por
semana.



Professores não
podem dar duas
matérias ao mesmo
tempo.



O professor precisa
estar disponível para
que possa dar aula
num horário.

Regras de preferência - espaçamento

	SEGUNDA	TERÇA	QUARTA	QUINTA	SEXTA
8h - 10h	MAC0425	MAC0425			MAC0315
10h - 12h	MAC0315				
14h - 16h	MAC0422		MAC0422		
16h - 18h					

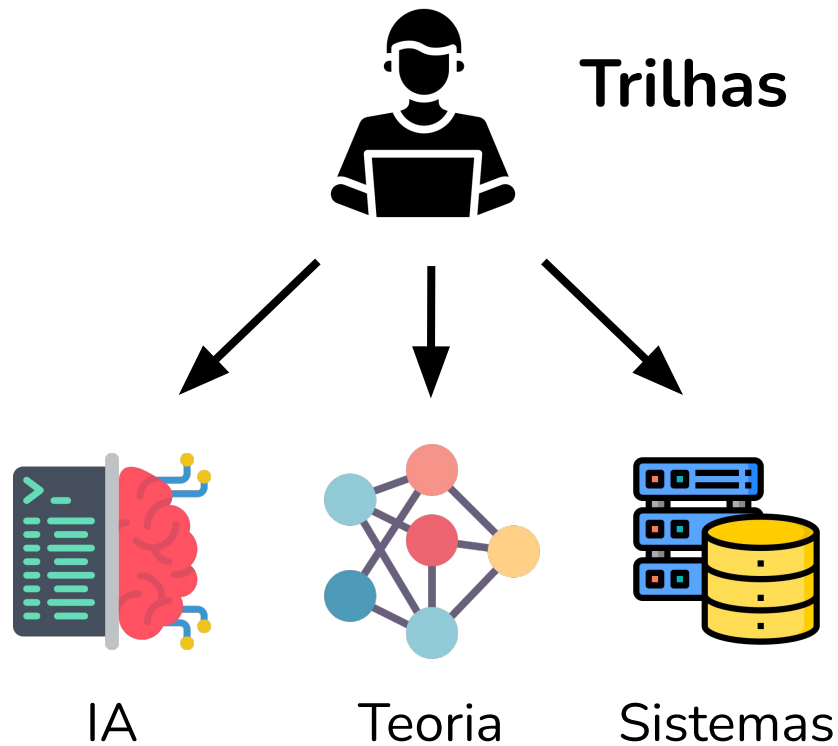
Levar em conta o espaçamento correto. De preferência, classes de uma matéria devem ser espaçadas com pelo menos um dia, mas também não devem estar distantes.

Regras de preferência - período

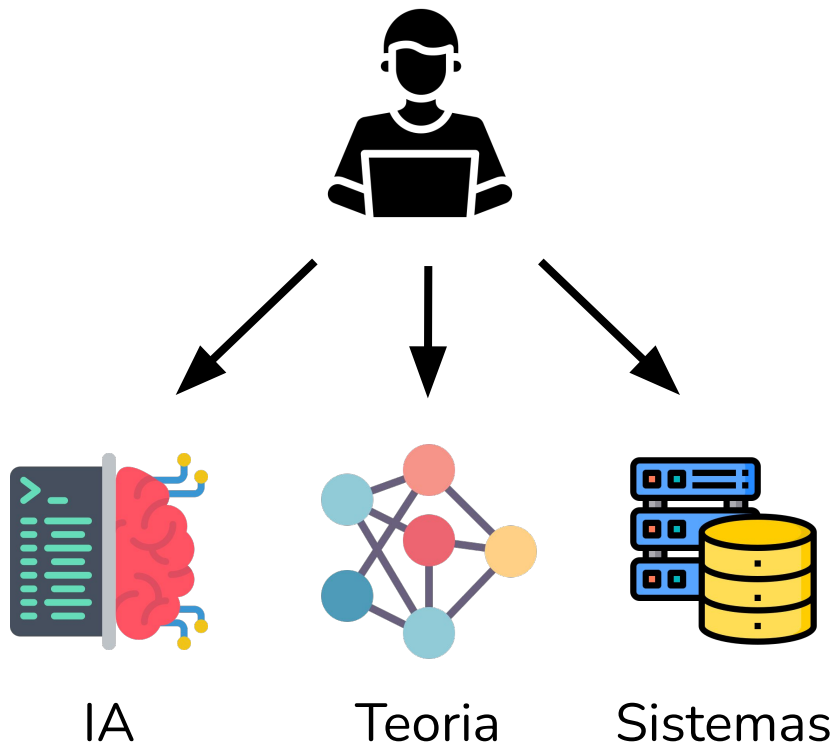
	SEGUNDA	TERÇA	QUARTA	QUINTA	SEXTA
8h - 10h	MAC0425				
10h - 12h	MAC0315		MAC0425		
14h - 16h			MAC0315		
16h - 18h					

De preferência, classes de uma matéria devem ser dadas no mesmo período.

O BCC possui algumas regras específicas



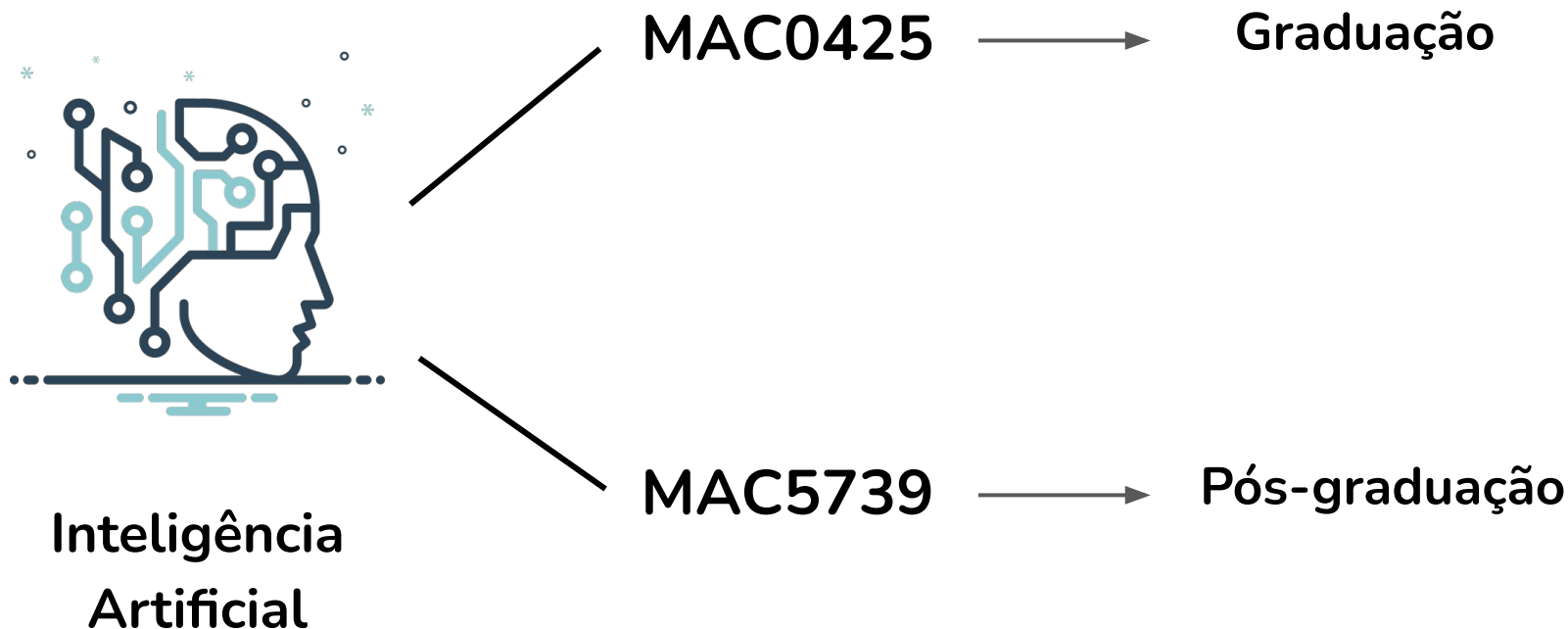
O BCC possui algumas regras específicas



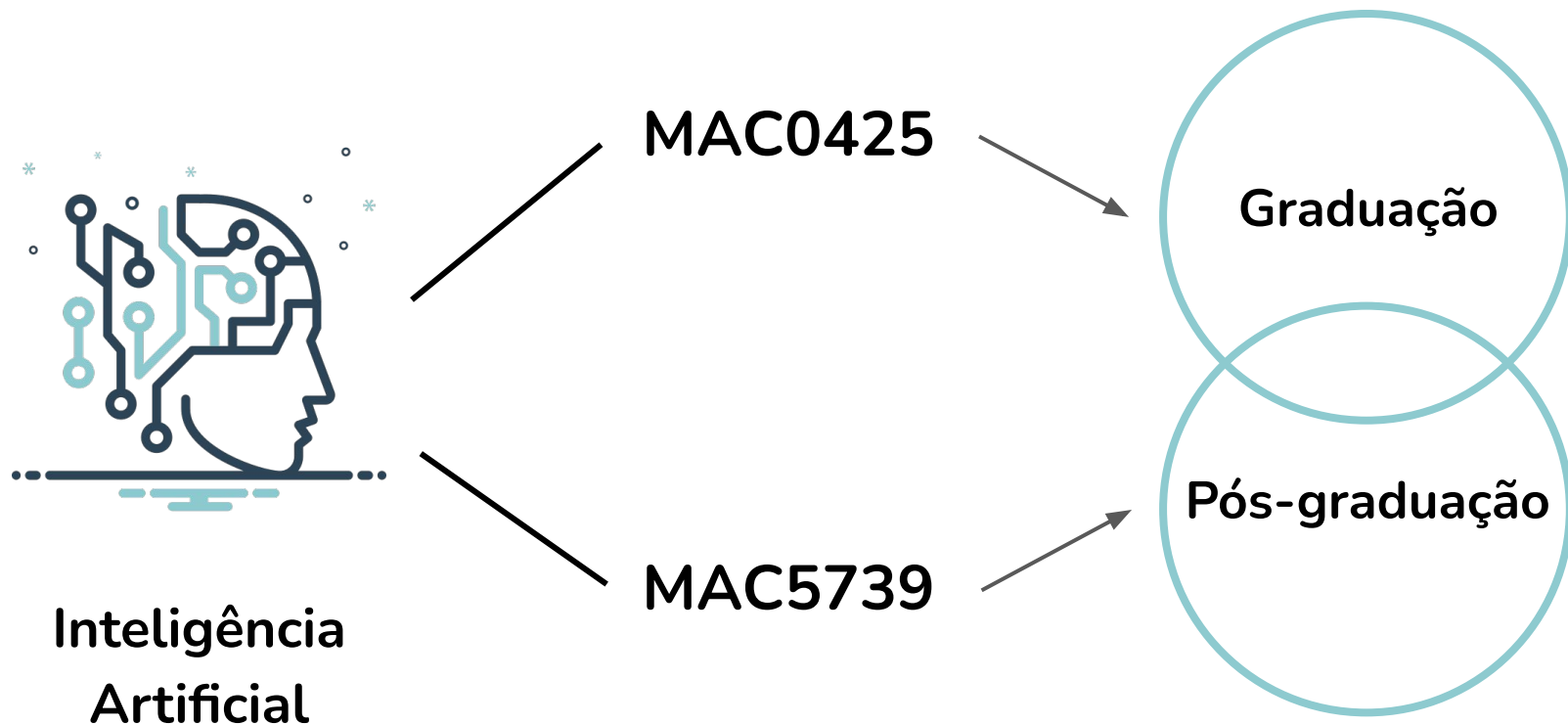
Trilhas

- Alunos tendem a se aprofundar em uma determinada área;
- Queremos **minimizar conflito entre matérias da mesma trilha** (graduação) ou escopo (pós-graduação).

Algumas disciplinas são dadas em conjunto



Algumas disciplinas são dadas em conjunto



O BCC possui algumas regras específicas

Matérias de estatística e ciências



O escalonador também precisa considerar...



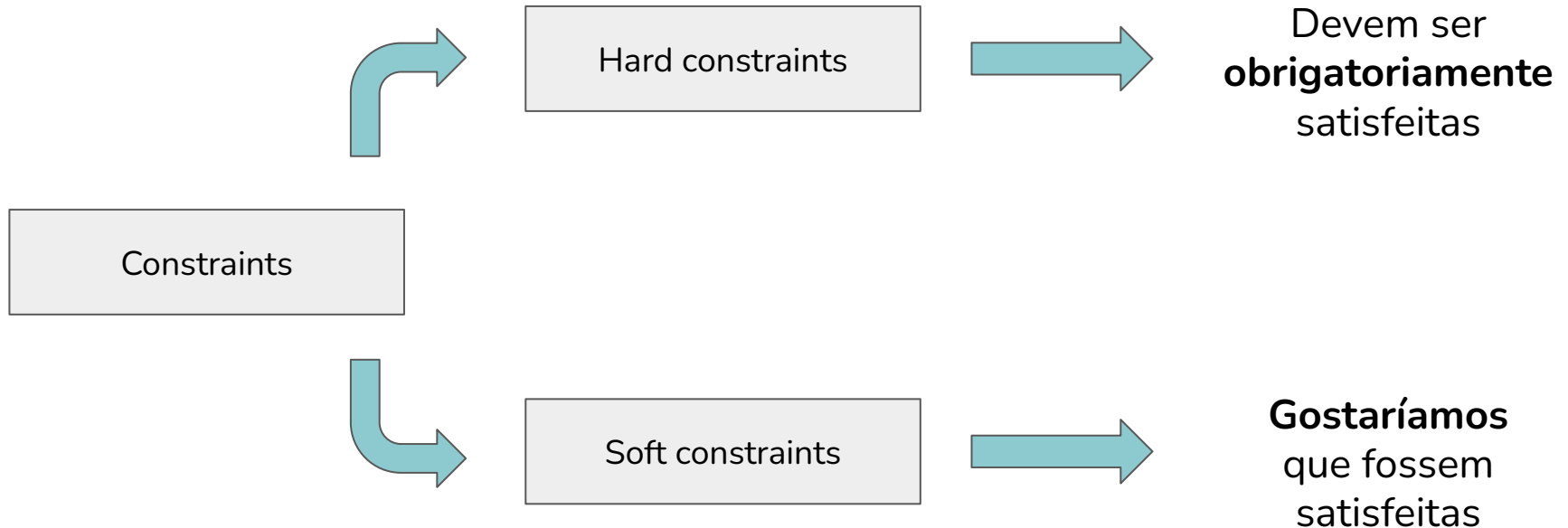
Sem aulas de sexta à tarde

Matérias dadas em outros departamentos



Obrigatórias do mesmo período não podem conflitar

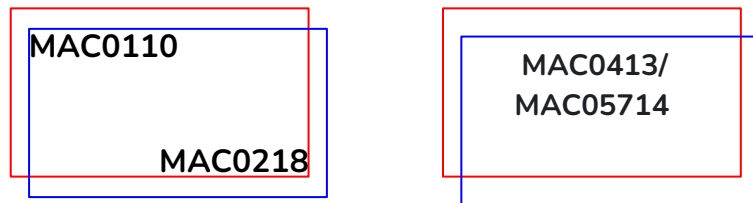
Sobre as Constraints



Exemplo: hard constraints

1

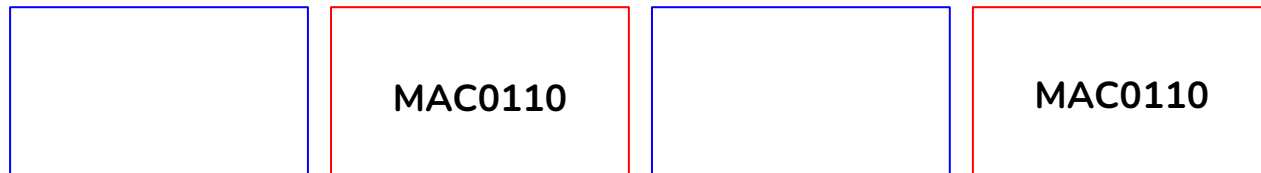
Duas disciplinas ministradas pelo mesmo professor não podem conflitar, a menos que sejam a mesma disciplina com siglas diferentes.



2

Aulas de uma mesma disciplina e turma não devem aparecer no mesmo dia.

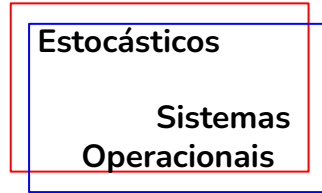
Segunda-feira



Exemplo: soft constraints

1

Disciplinas de estatística não devem conflitar com disciplinas obrigatórias a partir do segundo ano



Mecanismo geral do ASP

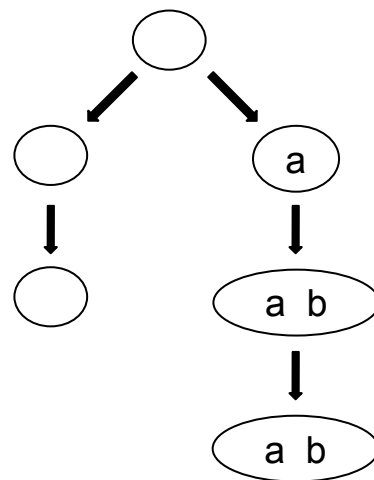
$\{a\}.$ \rightarrow choice rule

$b :- a.$ \rightarrow normal rule

$:- \text{not } b.$ \rightarrow constraint rule

átomo: a, b

literal: a, b, not b



Mecanismo geral do ASP

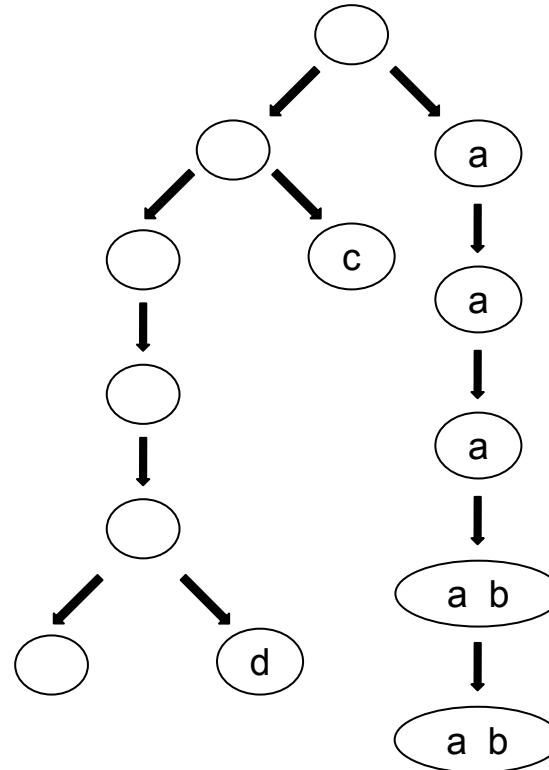
{a}.

{c} :- not a.

:- c, not a.

b :- a.

{d} :- not b, not c.



Hard constraints: representação em ASP

1

Duas disciplinas ministradas pelo mesmo professor não podem conflitar, a menos que sejam a mesma disciplina com siglas diferentes.

`:- course(C1, G1, T, _), course(C2, G2, T, _), conflict(C1, G1, C2, G2, _), not joint(C1,C2).`

2

Aulas de uma mesma disciplina e turma não devem aparecer no mesmo dia.

**`:- class(C, G, _, P1), class(C, G, _, P2),
P1 != P2,
P1/100 == P2/100,
not double(C).`**

Satisfatibilidade

O problema é **insatisfatível** quando não há uma forma de atribuir valores às variáveis de modo que nenhuma das hard constraints sejam violadas.

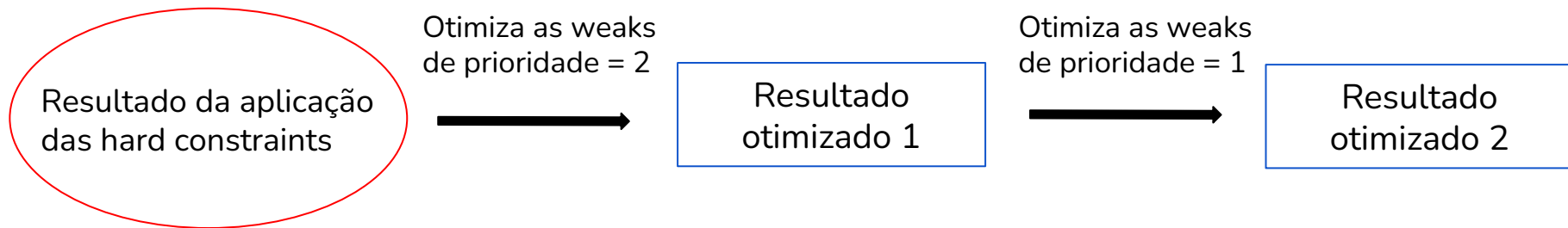
Da mesma forma, se há pelo menos uma forma de atribuir valores que não violem as constraints, então dizemos que o problema é **satisfatível**.

Soft constraint, prioridade e peso

Cada ocorrência de uma soft constraint é representada pelo predicado **Weak**(W, P)

Uma soft constraint é algo que **não** queremos que aconteça.

P = prioridade. Define a ordem com a qual as constraints são resolvidas.



E assim por diante...

Peso

Weak(W, P)

W = **peso**. É a penalidade; mede o quão ruim a ocorrência da soft constraint é.

Na nossa escala para prioridade = 1, **peso mínimo** = 5, **peso máximo** = 50.

50

20

10

5

"Super ruim"

"Muito ruim"

"Ruim"

"Um pouco ruim"

Otimização

Dados os pesos (penalidades) de cada soft constraint, o escalonador tenta minimizar a **soma desses pesos**.

Weak1(50, 1) vs. Weak2(20,1)



Weak2(20, 1)

Soft constraints: representação em ASP

1

Disciplinas de estatística não devem conflitar com disciplinas obrigatórias a partir do segundo ano

```
weak(W, Pr, Id) :- requiredElective(Cstat, statistics),  
                  obligatory(C2, X), X >= 3,  
                  conflict(Cstat, Gstat, C2, G2, _),  
                  W = @get_weight("sc5"),  
                  Pr = @get_priority("sc5"),  
                  Id = @concat("sc5", Cstat, Gstat, C2, G2).
```

Pergunta: o que é o "Id"?

Voltando à otimização...

Problema: cada ocorrência de uma soft constraint só é contabilizada uma vez.

Exemplo: disciplinas de estatística não devem conflitar com disciplinas obrigatórias a partir do segundo ano. **Prioridade** = 1, **peso** = 10.

Estocásticos conflita com Sistemas Operacionais



Weak1(10, 1)

Estocásticos conflita com Sistemas Operacionais

Estocásticos conflita com Modelagem e Simulação

Probabilidade I conflita com Sistemas Operacionais



Weak1(10, 1)

⋮

Voltando à otimização

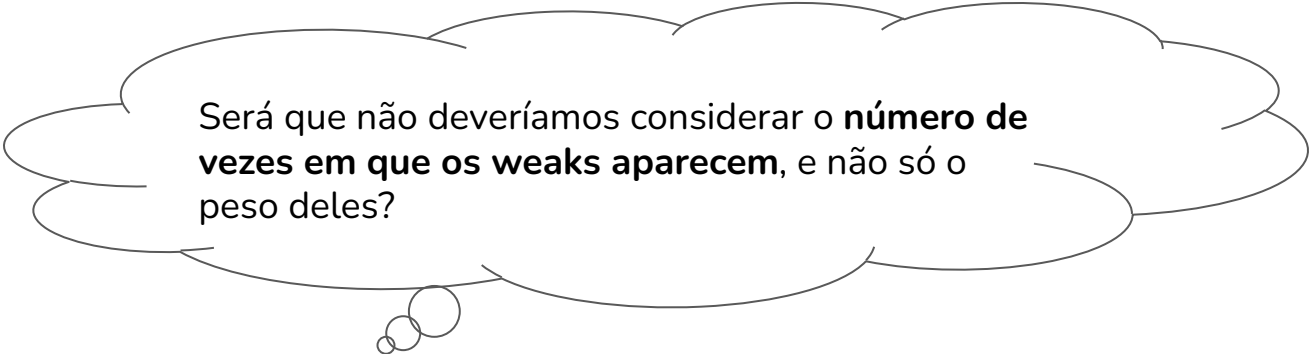
[1 ocorrência do tipo Weak1(50, 1)] vs. [1.000.000 de ocorrências do tipo Weak2(20, 1)]



Weak2(20, 1)

Vale mais a pena penalizar um Weak1(50, 1) do que um milhão de Weak2(20, 1) ?

Voltando à otimização...



Será que não deveríamos considerar o **número de vezes em que os weaks aparecem**, e não só o peso deles?

Contabilizando as ocorrências

Fizemos cada ocorrência gerar um Weak **único** através de um Id.

```
weak(W, Pr, Id) :- ...  
    ...,  
    Id = @concat("sc5", Cstat, Gstat, C2, G2).
```

Estocásticos conflita com Sistemas Operacionais

Estocásticos conflita com Modelagem e Simulação

Probabilidade I conflita com Sistemas Operacionais

⋮

Weak(**10**, 1, "sc5.MAE0228.45.MAC0422.45")

Weak(**10**, 1, "sc5.MAE0228.45.MAC0209.45")

Weak(**10**, 1, "sc5.MAE0127.45.MAC0422.45")

Otimizando

- 1 Agrupa as Weaks por peso

count(W, P, S) :- S = #count{ C : weak(W, P, C)}, weak(W, P, _).

- 2 Define **W** de uma Weak. W é o peso da Weak multiplicado pelo nº de ocorrências dela.

W(Wnew, P) :- count(W,P,S), Wnew = W*S.

- 3 Minimiza a soma dos pesos.

#minimize { W@P : w(W,P) }.

Integração Clingo - Python

Operações no Clingo são limitadas

Exemplo: Lógica condicional (*if / else*)

É possível criar funções em Python
que devolvem valores
compreendidos pelo Clingo

```
weak(W, Pr, Id) :- class(C1, G1, _, P), class(C2, G2, _, P),
    conflict(C1, G1, C2, G2, P),
    obligatory(C1, IDEAL1), elective(C2, IDEAL2),
    W = @calculate_weight_sc1(IDEAL1, IDEAL2),
    Pr = @get_priority("sc1"),
    Id = @concat("sc1",C1,G1,C2,G2).
```

```
#script(python)
import clingo

def calculate_weight_sc1(IDEAL1, IDEAL2):
    base_weight = get_weight("sc1")
    period1 = int(str(IDEAL1))
    period2 = int(str(IDEAL2))
    difference = abs(period1 - period2)
    if difference == 0:
        weight = base_weight
    elif difference > 5:
        weight = 0
    else:
        weight = base_weight//((difference * 2))
    return clingo.Number(weight)

def get_soft_name(Id):
    Id = str(Id).replace("'", '').replace('"', "")
    return Id.split(".")[0]

def joint_id(A,B):
    lista = [A,B]
    lista.sort()
    return "joint" + str(lista[0]) + str(lista[1])

#end.
```

Integração Clingo - Python: outras utilidades

Essencial para criarmos um “Id” para cada *soft constraint*

- Em geral, predicado só é contabilizado 1 vez
- Criação de “Id” permite contabilizar ocorrências de *softs constraints*
- Python permite operação de concatenação de *strings*

```
#script(python)

def concat(*args):
    ans = ""
    for i in args:
        s = str(i)
        s = s.replace("'", '').replace('"', '')
        ans+= str(s) + "."
    return ans[0:-1]

#end.
```

Integração Clingo - Python: outras utilidades

Essencial para criarmos a configuração de pesos

- Usuário altera o peso/prioridade de cada *constraint*

```
#script(python)
DEFAULT_PRIORITY = 1

def w(weight, priority=DEFAULT_PRIORITY):
    "Returns a dictionary with the corresponding weight and priority."
    return {"weight": weight, "priority": priority}

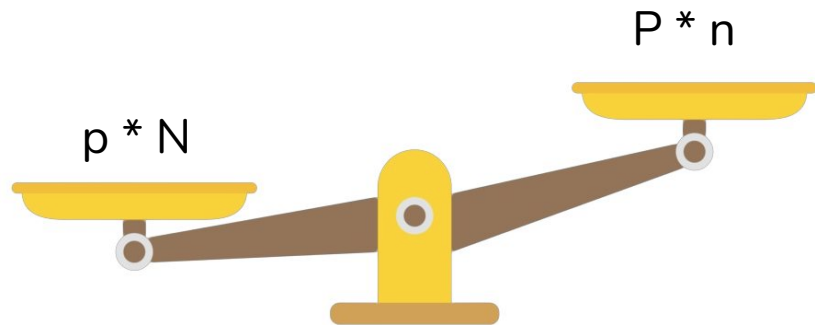
SOFT_CONSTRAINTS_WEIGHTS = {
    "sc1": w(20),
    "sc2": w(20),
    "sc3": w(10),
    "sc4": w(5),
    "sc5": w(10),
    "sc6": w(5),
    "sc8": w(25),
    "sc9": w(10),
    "sc10": w(10),
    "sc12": w(10),
    "sc13": w(50),
    "sc14": w(5, 0),
    "sc15": w(1, -1),
}
#end.
```


Desafio: como encontrar os pesos ideais?

O que é um peso ideal?

Reflete a importância real da regra

Penalidade = Peso * (Nº de ocorrências)



Modos de encontrar o peso ideal:

- Tentativa e erro
- Separar *constraints* semelhantes em prioridades iguais



Problema:

Minimiza MUITO a importância das regras menos prioritárias; Nº de ocorrências não importa.

Solução encontrada: *sc_metrics.lp*

Criação de um contador explícito de ocorrências

Muito útil durante o desenvolvimento

Solução final: Tentativa e erro + *sc_metrics.lp*

```
%*
weak/4(weight, priority, id, soft name)
-----
*%
weak(W, P, Id, Soft) :- weak(W, P, Id), Soft = @get_soft_name(Id).

%*
scCount/2(soft name, quantity)
-----
*%
scCount(Soft,C) :- C = #count{ W,P,Id : weak(W,P,Id,Soft)}, weak(_,_,_,Soft).
```

Métricas

1

Nº de ocorrências de *soft constraints*

A ocorrência de uma *soft constraint* implica uma penalidade de peso W .

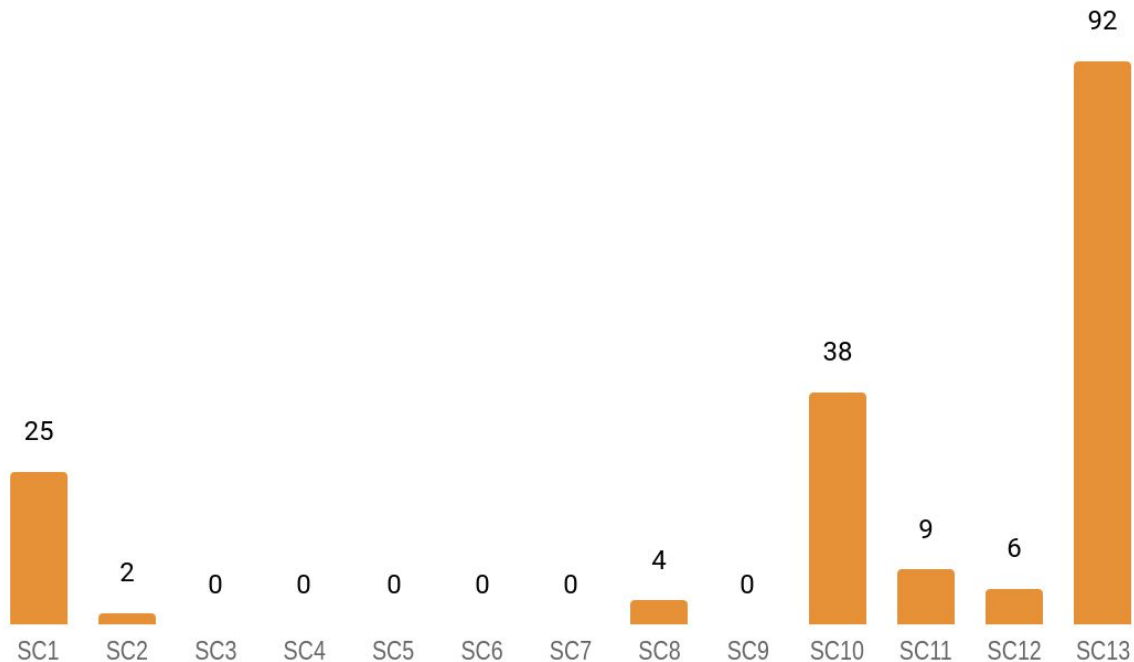
2

Otimização

É o somatório de todas as penalidades.

Grade real: n° de ocorrências de soft constraints

OCORRÊNCIAS



Escalonador: n° de ocorrências de soft constraints



Otimização grade real vs. Otimização Escalonador

Grade real

Soft 1 - 9 ,11 e 12	1085
Soft 10	190
Soft 13	92
Total	1367

Escalonador

Soft 1 - 9, 11 e 12	110
Soft 10	215
Soft 13	100
Total	425

Métricas: conclusões

Nº de otimização da grade real é
vezes maior que o do programa

$$1367 \div 425 = 3,2$$

O que isso significa?

- Com o nosso modelo (*inputs*, regras, pesos e prioridades), o escalonamento gerado é mais desejável

Métricas: conclusões

SC10: classes devem ocorrer nos horários preferenciais de cada professor

SC13: qualquer tipo de conflito deve ser evitado

Nosso escalonamento é um pouco pior para a SC10 (5 ocorrências a mais) e SC13 (8 ocorrências a mais)

No resto, nosso escalonamento é **10** vezes melhor

Desempenho

Resultado não é ótimo → Programa não termina por conta própria



É necessário limitar o tempo de execução

Qual o melhor limite de tempo? (melhor “custo-benefício”)

R: **500** segundos

Observação: o desempenho depende bastante do *hardware*

Gráfico de desempenho



Solução:

Optimization: 110 215 100

Horário	Segunda	Terça	Quarta	Quinta	Sexta
08:00-09:40	mac0110-bcc mac6992-bcc_pos ibi5092-bcc_pos mac0691-bcc mac6918-bcc_pos	mac0329-bcc mac0210-bcc mac6931-bcc_pos	mac0422-bcc mac0327-bcc mac5753-bcc_pos mac0425-bcc mac5739-bcc_pos mac0417-bcc mac5768-bcc_pos mac0420-bcc mac5744-bcc_pos	mac0323-bcc mac0350-bcc ibi5037-bcc_pos	mac0105-bcc mac6711-bcc_pos
10:00-11:40	mac0422-bcc ibi5031-bcc_pos mac0327-bcc mac5753-bcc_pos	mac0323-bcc mac0350-bcc ibi5037-bcc_pos mac0467-bcc mac6909-bcc_pos	mac0110-bcc mac6711-bcc_pos ibi5031-bcc_pos mac6992-bcc_pos ibi5092-bcc_pos mac0691-bcc mac6918-bcc_pos mac0436-bcc	mac0329-bcc mac0210-bcc mac0467-bcc mac6909-bcc_pos mac6931-bcc_pos	mac0425-bcc mac5739-bcc_pos mac0417-bcc mac5768-bcc_pos mac0420-bcc mac5744-bcc_pos mac0436-bcc
14:00-15:40	mac0209-bcc mac0439-bcc	mac0101-bcc mac4722-bcc_pos	mac0105-bcc mac6956-bcc_pos mac6937-bcc_pos	mac0427-bcc mac0102-bcc mac4722-bcc_pos	mac6956-bcc_pos mac6937-bcc_pos
16:00-17:40	mac0427-bcc mac0209-bcc mac0219-bcc mac5742-bcc_pos	mac0345-bcc mac0320-bcc mac5770-bcc_pos mac0473-bcc mac5780-bcc_pos mat0223-bcc mac0218-bcc mac0460-bcc mac5832-bcc_pos mac0336-bcc mac5723-bcc_pos	mac0219-bcc mac5742-bcc_pos mac0439-bcc	mac0345-bcc mac0320-bcc mac5770-bcc_pos mac0473-bcc mac5780-bcc_pos mat0223-bcc mac0218-bcc mac0460-bcc mac5832-bcc_pos mac0336-bcc mac5723-bcc_pos	mac6989-bcc_pos

- programa gera várias soluções como essa (terminal e csv);

- usuário define tempo de execução (+ tempo = + otimização);

- pesos facilmente configuráveis;

Nossa solução é realmente boa?



Vantagens

- Bom desempenho, se comparado com a grade manual;
- Possibilita mudar algumas entradas e rodar de novo rapidamente;
- Expansível para outros cursos e configurável;



Desvantagens

- Maior esforço para definir o input;
- Expansões/modificações requerem que o cliente saiba escrever código em ASP;

Conclusão: Escalonamento

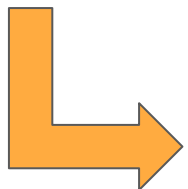
Programação lógica permite criar um mundo próprio, com regras próprias, etc.

Clingo + ASP é ideal para resolver o problema

- Mecanismo de otimização é a chave para uma boa solução

Vantagens superam as desvantagens

O BCC é um curso com muitas particularidades



Quanto mais “estático” o curso for, mais fácil é a modelagem. Várias regras do BCC também podem ser aplicadas em outros cursos.

Conclusão: Experiência

Dificuldades:

- Aprender linguagem nova
- Lidar com *inputs* (muitas vezes) irregulares
- Modelar particularidades do curso
- Encontrar os pesos ideais

Esperamos que possa ser útil para anos futuros

Possibilidade de evolução: permitir alterações após a geração da solução final, de acordo com o *feedback* dos alunos e professores

Hard constraints:

i Constraint

- 1 Um professor não pode dar duas matérias ao mesmo tempo
- 2 Todas as matérias propostas para o semestre devem ser dadas
- 3 Cada matéria tem uma quantidade definida de aulas semanais
- 4 As aulas devem ser dadas nos horários disponíveis dos professores
- 5 Graduação não pode ter aulas de sexta-feira a tarde
- 6 **Obrigatórias** não podem conflitar com outras **obrigatórias** do mesmo ano
- 7 Existem horários constantes para disciplinas do primeiro ano e segundo
- 8 Existem matérias que devem ter seus dois horários juntos (teoria + prática)
- 9 Não ter horários de uma mesma disciplina no mesmo dia (exceto algumas matérias)
- 10 Classes com a mesma sigla devem ser dadas no mesmo horário

Soft constraints:

i Constraint

- 1 Obrigatórias não podem conflitar com eletivas de período próximos
- 2 Obrigatórias não podem conflitar com obrigatórias de anos diferentes
Obrigatórias de trilhas não podem conflitar com outras matérias da mesma
- 3 trilha
Não-obrigatórias de trilhas não podem conflitar com outras
- 4 não-obrigatórias da mesma trilha
- 5 Não conflitar matérias de ciências com obrigatórias
Não conflitar matérias de estatísticas com obrigatórias a partir do segundo
- 6 ano
Não conflitar matérias dos blocos de Sistema e Teoria (Teoria com maior
- 7 peso)
- 8 Diminuir o conflito com disciplinas de alta demanda
- 9 Evitar conflito entre matérias do mesmo escopo
- 10 Horários que os professores preferem
- 11 Não ter horários de uma mesma disciplina em dias seguidos
- 12 Não ter horários de uma mesma disciplina em períodos diferentes
- 13 Evitar todo tipo de conflito

Repositório

<https://github.com/anayflima/class-scheduler>

Referências

https://pt.wikipedia.org/wiki/Programa%C3%A7%C3%A3o_declarativa

<https://edisciplinas.usp.br/>

<https://uspdigital.usp.br/jupiterweb/obterDisciplina?nomdis=&sgldis=mac0472>

<https://pt.vecteezy.com/png/9663881-escala-peso-icone-png-transparente>

Referências Ícones

slide 3:

https://www.flaticon.com/free-icon/graduated_3135773?term=student&page=1&position=1&origin=search&related_id=3135773

https://www.flaticon.com/free-icon/graduated_3135773?term=student&page=1&position=1&origin=search&related_id=3135773

https://www.flaticon.com/free-icon/graduated_3135773?term=student&page=1&position=1&origin=search&related_id=3135773

slide 7:

https://www.flaticon.com/free-icon/schedule_3269690?term=calendar&page=1&position=90&origin=search&related_id=3269690

slide 8

https://www.flaticon.com/free-icon/calendar_2838779?term=calendar&page=1&position=1&origin=search&related_id=2838779

https://www.flaticon.com/br/icone-gratis/reuniao_4415132?term=reuniao&related_id=4415132

slide 37:

https://www.flaticon.com/br/icone-gratis/tabela_178916?term=tabela&page=1&position=2&origin=search&related_id=178916

https://www.flaticon.com/br/icone-gratis/internet_4386645?term=internet&page=1&position=1&origin=search&related_id=4386645

https://www.flaticon.com/free-icon/conversation_3050431?term=meet&page=1&position=1&origin=search&related_id=3050431