Ganhar US$ 1.000.000,00 só jogando Mario

Érik Amorim

(ICMC - USP São Carlos)

Novembro de 2013

Intro

Classic Nintendo games are (NP-)hard
(03/2012, arXiv)

Intro

Classic Nintendo games are (NP-)hard
(03/2012, arXiv)

- Greg Aloupis (Université Libre de Bruxelles)
- Erik D. Demaine (Massachussets Institute of Technology)
- Alan Guo (Massachussets Institute of Technology)

Intro

Classic Nintendo games are (NP-)hard
(03/2012, arXiv)

- Greg Aloupis (Université Libre de Bruxelles)
- Erik D. Demaine (Massachussets Institute of Technology)
- Alan Guo (Massachussets Institute of Technology)

All products, company names, brand names, trademarks, and sprites are properties of their respective owners. Sprites are used here under the Fair Use for the educational purpose of illustrating mathematical theorems.

Intro

Objetivo: Provar que

Intro

Objetivo: Provar que

- Super Mario Bros. 1, 3
- Super Mario World
- Donkey Kong Country 1, 2, 3
- Legend of Zelda
- Metroid
- Pokémon Blue, Red, Yellow, Gold, Silver, Ruby, Sapphire...

Intro

Objetivo: Provar que

- Super Mario Bros. 1, 3
- Super Mario World
- Donkey Kong Country 1, 2, 3
- Legend of Zelda
- Metroid
- Pokémon Blue, Red, Yellow, Gold, Silver, Ruby, Sapphire...

são difíceis.

Intro

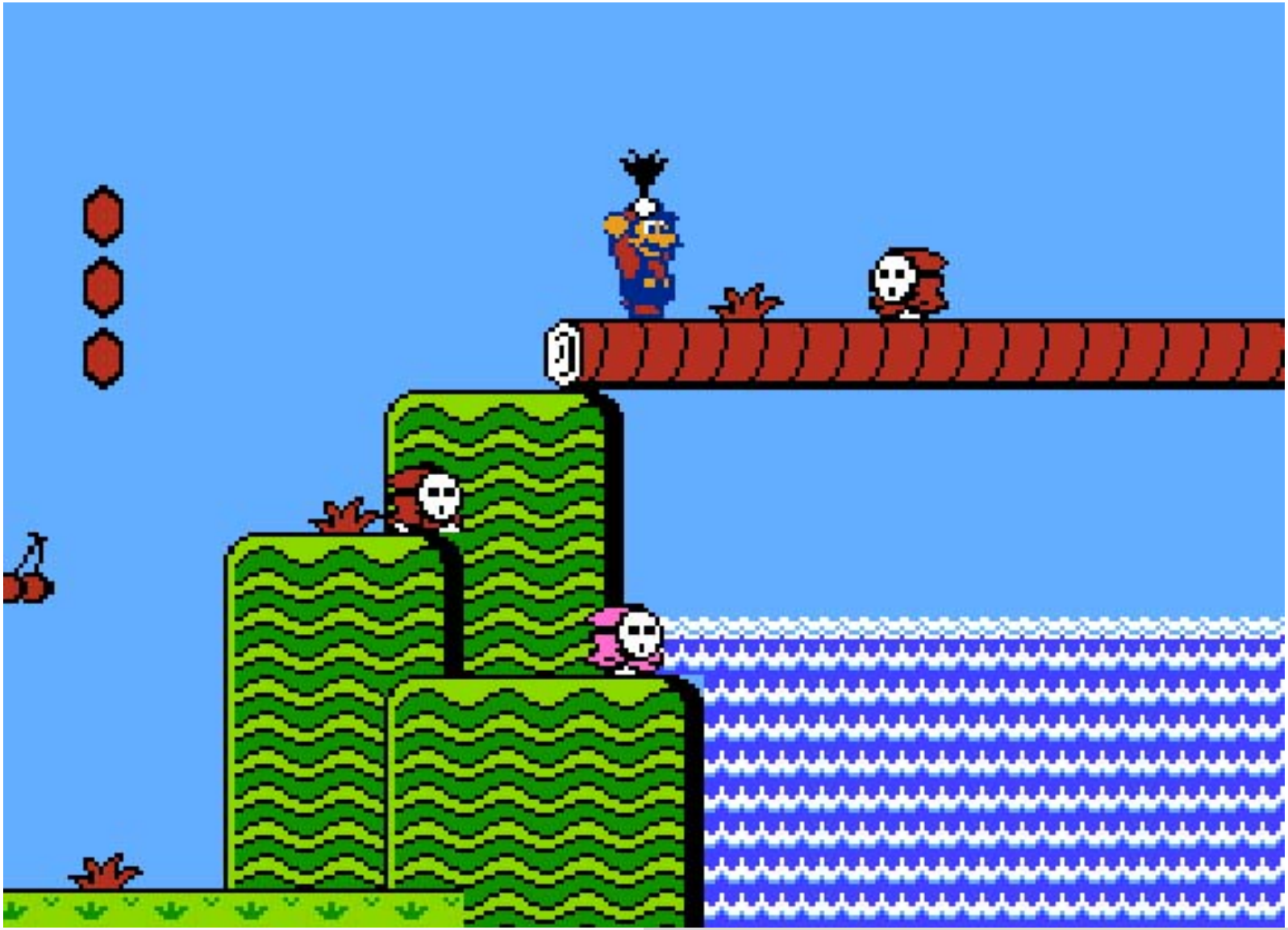
Objetivo: Provar que

- Super Mario Bros. 1, 3
- Super Mario World
- Donkey Kong Country 1, 2, 3
- Legend of Zelda
- Metroid
- Pokémon Blue, Red, Yellow, Gold, Silver, Ruby, Sapphire...

são difíceis. (Num certo sentido)

Super Mario Bros.

Super Mario Bros. 2



Super Mario Bros. 3

Super Mario World

Problemas de decisão

Um problema de decisão é uma pergunta com resposta sim ou não.

Problemas de decisão

Um problema de decisão é uma pergunta com resposta sim ou não.

- *Existe um número perfeito ímpar?*

Problemas de decisão

Um problema de decisão é uma pergunta com resposta sim ou não.

- *Existe um número perfeito ímpar?* ✓

Problemas de decisão

Um problema de decisão é uma pergunta com resposta sim ou não.

- *Existe um número perfeito ímpar?* ✓
- *Que horas são?*

Problemas de decisão

Um problema de decisão é uma pergunta com resposta sim ou não.

- *Existe um número perfeito ímpar?* ✓
- *Que horas são?* ✗

Problemas de decisão

Um problema de decisão é uma pergunta com resposta sim ou não.

- *Existe um número perfeito ímpar?* ✓
- *Que horas são?* ✗
- *A resposta desse problema é não?*

Problemas de decisão

Um problema de decisão é uma pergunta com resposta sim ou não.

- *Existe um número perfeito ímpar?* ✓
- *Que horas são?* ✗
- *A resposta desse problema é não?* ✓

Problemas de decisão

Um problema de decisão é uma pergunta com resposta sim ou não.

- *Existe um número perfeito ímpar?* ✓
- *Que horas são?* ✗
- *A resposta desse problema é não?* ✓

Alguns problemas dependem de uma entrada.

Problemas de decisão

Problema

2013 *é primo?*

Problemas de decisão

Problema

2013 *é primo?*

Resposta: Não.

Problemas de decisão

Problema

2013 *é primo?*

Resposta: Não.

Problema

Um inteiro positivo n é primo?

Problemas de decisão

Problema

2013 *é primo?*

Resposta: Não.

Problema

Um inteiro positivo n é primo?

Resposta: Depende.

Problemas de decisão

Problema

2013 *é primo?*

Resposta: Não.

Problema

Um inteiro positivo n é primo?

Resposta: Depende.
Algoritmo: Entrada n

Problemas de decisão

Problema

2013 *é primo?*

Resposta: Não.

Problema

Um inteiro positivo n é primo?

Resposta: Depende.
Algoritmo: Entrada n

- Se $n = 1$, esquece.

Problemas de decisão

Problema

2013 *é primo?*

Resposta: Não.

Problema

Um inteiro positivo n é primo?

Resposta: Depende.
Algoritmo: Entrada n

- Se $n = 1$, esquece.
- Tentar dividir n por $2, 3, 4, 5, \ldots, n - 1$.

Problemas de decisão

Problema

2013 *é primo?*

Resposta: Não.

Problema

Um inteiro positivo n é primo?

Resposta: Depende.
Algoritmo: Entrada n

- Se $n = 1$, esquece.
- Tentar dividir n por $2, 3, 4, 5, \ldots, n - 1$.
- Se deu certo com algum número, n é composto.

Problemas de decisão

Problema

2013 *é primo?*

Resposta: Não.

Problema

Um inteiro positivo n é primo?

Resposta: Depende.
Algoritmo: Entrada n

- Se $n = 1$, esquece.
- Tentar dividir n por $2, 3, 4, 5, \ldots, n - 1$.
- Se deu certo com algum número, n é composto.
- Se não deu certo com nenhum número, n é primo.

Tempo polinomial

Para problemas de decisão que recebem entradas, um algoritmo que encontra a resposta vai demorar algum tempo para fazer isso.

Tempo polinomial

Para problemas de decisão que recebem entradas, um algoritmo que encontra a resposta vai demorar algum tempo para fazer isso. O tempo é maior quanto maior for o tamanho da entrada n.

Tempo polinomial

Para problemas de decisão que recebem entradas, um algoritmo que encontra a resposta vai demorar algum tempo para fazer isso. O tempo é maior quanto maior for o tamanho da entrada n.

Formatos não usados	**Formatos usados**
1089	1
$5n^4 - 8n^2 + 7, 3n - 3, 14$	$n^{0.01}$, $\sqrt{n}$, n, n^2, $n^{\frac{3}{2}}$, n^3
$2(\log n)^3 - 5n(\log n)$	$\log n$, $(\log n)^2$, $n \log n$, $n^{9.93}(\log n)^{4.02}$
45×3^{8n}	e^n, 2^n, 0.0001^n, 999^n
$10n!$	$n!$

Comportamento assintótico

- 1 é mais rápido que qualquer coisa

Comportamento assintótico

- 1 é mais rápido que qualquer coisa
- n^α é mais rápido que n^β se e somente se $\alpha < \beta$

Comportamento assintótico

- 1 é mais rápido que qualquer coisa
- n^α é mais rápido que n^β se e somente se $\alpha < \beta$
- $\log n$ é mais rápido que n^α para qualquer $\alpha > 0$

Comportamento assintótico

- 1 é mais rápido que qualquer coisa
- n^α é mais rápido que n^β se e somente se $\alpha < \beta$
- $\log n$ é mais rápido que n^α para qualquer $\alpha > 0$
- n^α é mais rápido que γ^n para qualquer $\alpha > 0$ e qualquer $\gamma > 1$

Comportamento assintótico

- 1 é mais rápido que qualquer coisa
- n^α é mais rápido que n^β se e somente se $\alpha < \beta$
- $\log n$ é mais rápido que n^α para qualquer $\alpha > 0$
- n^α é mais rápido que γ^n para qualquer $\alpha > 0$ e qualquer $\gamma > 1$
- Qualquer coisa é mais rápida que $n!$

Comportamento assintótico

- 1 é mais rápido que qualquer coisa
- n^α é mais rápido que n^β se e somente se $\alpha < \beta$
- $\log n$ é mais rápido que n^α para qualquer $\alpha > 0$
- n^α é mais rápido que γ^n para qualquer $\alpha > 0$ e qualquer $\gamma > 1$
- Qualquer coisa é mais rápida que $n!$

$\therefore$ Tempo polinomial é melhor que exponencial

Ordenação

Problema

Toda lista finita de números inteiros pode ser ordenada?

Ordenação

Problema

Toda lista finita de números inteiros pode ser ordenada?

Resposta: Sim.

Ordenação

Problema

Toda lista finita de números inteiros pode ser ordenada?

Resposta: Sim.
Algoritmos: (Entrada é uma lista de tamanho n)

Ordenação

Problema

Toda lista finita de números inteiros pode ser ordenada?

Resposta: Sim.

Algoritmos: (Entrada é uma lista de tamanho n)

- *Quicksort*, *Merge Sort*, *Heapsort*: $n \log n$

Ordenação

Problema

Toda lista finita de números inteiros pode ser ordenada?

Resposta: Sim.

Algoritmos: (Entrada é uma lista de tamanho n)

- *Quicksort*, *Merge Sort*, *Heapsort*: $n \log n$
- *Shell Sort*: $n^{\frac{3}{2}}$

Ordenação

Problema

Toda lista finita de números inteiros pode ser ordenada?

Resposta: Sim.

Algoritmos: (Entrada é uma lista de tamanho n)

- *Quicksort*, *Merge Sort*, *Heapsort*: $n \log n$
- *Shell Sort*: $n^{\frac{3}{2}}$
- *Bubble Sort*: n^2

Ordenação

Problema

Toda lista finita de números inteiros pode ser ordenada?

Resposta: Sim.

Algoritmos: (Entrada é uma lista de tamanho n)

- *Quicksort*, *Merge Sort*, *Heapsort*: $n \log n$
- *Shell Sort*: $n^{\frac{3}{2}}$
- *Bubble Sort*: n^2
- *Bogosort*: $n \cdot n!$

Ordenação

Problema

Toda lista finita de números inteiros pode ser ordenada?

Resposta: Sim.

Algoritmos: (Entrada é uma lista de tamanho n)

- *Quicksort*, *Merge Sort*, *Heapsort*: $n \log n$
- *Shell Sort*: $n^{\frac{3}{2}}$
- *Bubble Sort*: n^2
- *Bogosort*: $n \cdot n!$
- *Inteligent Design Sort*: 0

P=NP?

Classe P: É a classe dos problemas de decisão que podem ser resolvidos em tempo polinomial.

P=NP?

Classe P: É a classe dos problemas de decisão que podem ser resolvidos em tempo polinomial.
Classe NP: É a classe dos problemas de decisão

P=NP?

Classe P: É a classe dos problemas de decisão que podem ser resolvidos em tempo polinomial.
Classe NP: É a classe dos problemas de decisão para os quais é possível verificar em tempo polinomial se um certo candidato a resposta é de fato uma resposta.

P=NP?

Classe P: É a classe dos problemas de decisão que podem ser resolvidos em tempo polinomial.
Classe NP: É a classe dos problemas de decisão para os quais é possível verificar em tempo polinomial se um certo candidato a resposta é de fato uma resposta.

- Um problema em P é um pepino que pode ser resolvido rapidamente.

P=NP?

Classe P: É a classe dos problemas de decisão que podem ser resolvidos em tempo polinomial.
Classe NP: É a classe dos problemas de decisão para os quais é possível verificar em tempo polinomial se um certo candidato a resposta é de fato uma resposta.

- Um problema em P é um pepino que pode ser resolvido rapidamente.
- Um problema em NP é um pepino que você talvez não saiba resolver,

P=NP?

Classe P: É a classe dos problemas de decisão que podem ser resolvidos em tempo polinomial.
Classe NP: É a classe dos problemas de decisão para os quais é possível verificar em tempo polinomial se um certo candidato a resposta é de fato uma resposta.

- Um problema em P é um pepino que pode ser resolvido rapidamente.
- Um problema em NP é um pepino que você talvez não saiba resolver, mas você sabe rapidamente dar pitaco na solução dos outros, ou concordar com ela.

P=NP?

Classe P: É a classe dos problemas de decisão que podem ser resolvidos em tempo polinomial.
Classe NP: É a classe dos problemas de decisão para os quais é possível verificar em tempo polinomial se um certo candidato a resposta é de fato uma resposta.

- Um problema em P é um pepino que pode ser resolvido rapidamente.
- Um problema em NP é um pepino que você talvez não saiba resolver, mas você sabe rapidamente dar pitaco na solução dos outros, ou concordar com ela.

$P \subseteq NP$.

P=NP?

Classe P: É a classe dos problemas de decisão que podem ser resolvidos em tempo polinomial.
Classe NP: É a classe dos problemas de decisão para os quais é possível verificar em tempo polinomial se um certo candidato a resposta é de fato uma resposta.

- Um problema em P é um pepino que pode ser resolvido rapidamente.
- Um problema em NP é um pepino que você talvez não saiba resolver, mas você sabe rapidamente dar pitaco na solução dos outros, ou concordar com ela.

$P \subseteq NP$. Mas será que $P = NP$?

P=NP?

Classe P: É a classe dos problemas de decisão que podem ser resolvidos em tempo polinomial.
Classe NP: É a classe dos problemas de decisão para os quais é possível verificar em tempo polinomial se um certo candidato a resposta é de fato uma resposta.

- Um problema em P é um pepino que pode ser resolvido rapidamente.
- Um problema em NP é um pepino que você talvez não saiba resolver, mas você sabe rapidamente dar pitaco na solução dos outros, ou concordar com ela.

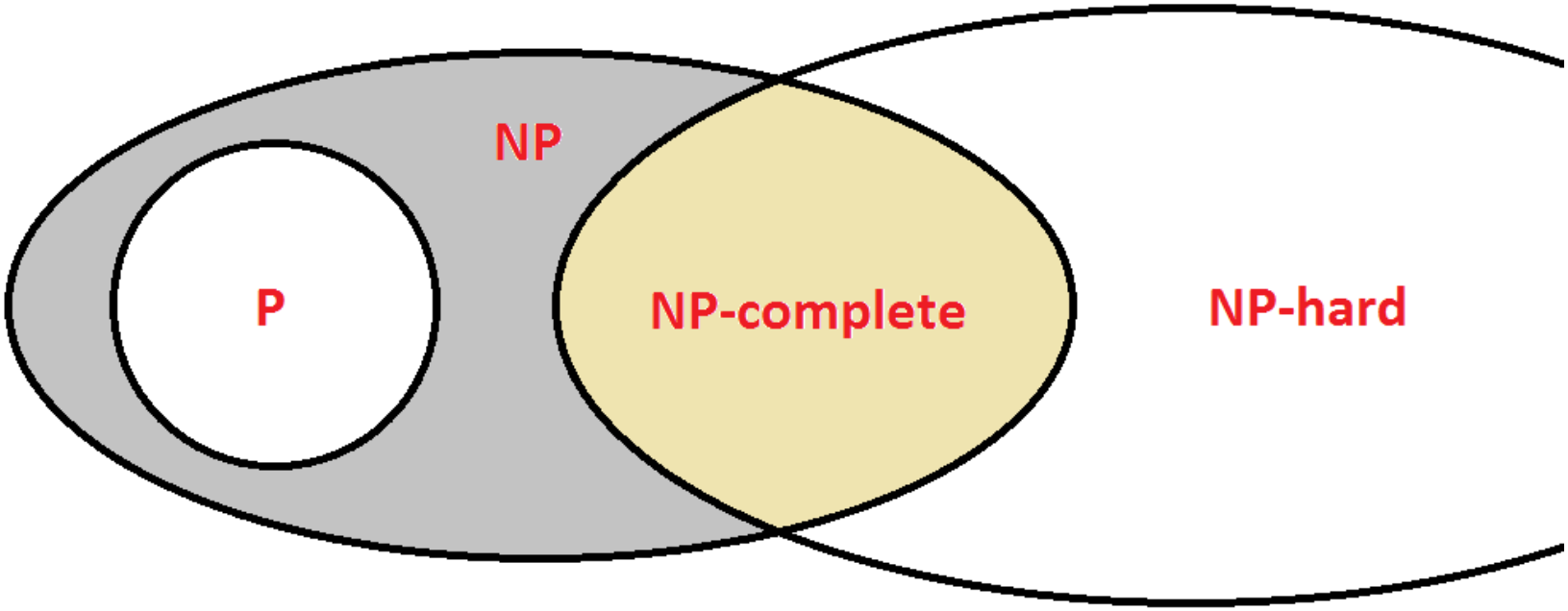
$P \subseteq NP$. Mas será que $P = NP$? Esse é um dos 7 *problemas do milênio*.

Citação

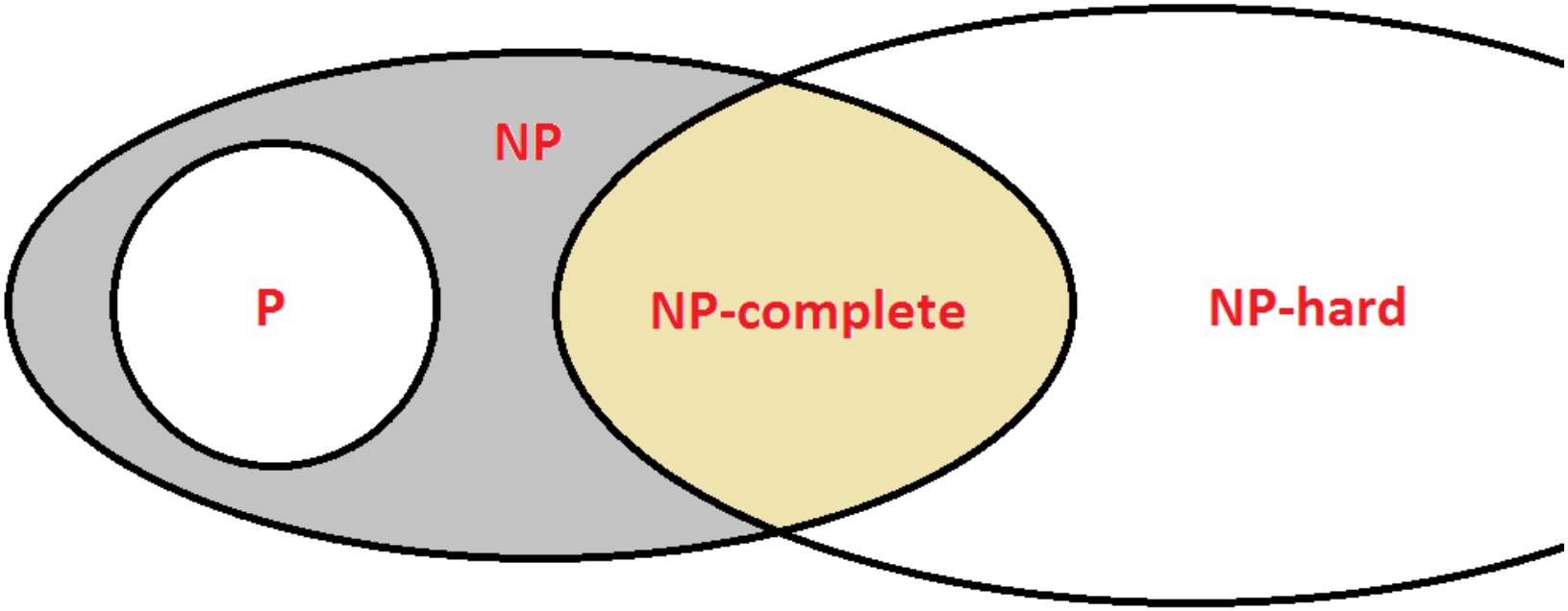
"If P = NP, then the world would be a profoundly different place than we usually assume it to be. There would be no special value in creative leaps, no fundamental gap between solving a problem and recognizing the solution once it's found. Everyone who could appreciate a symphony would be Mozart; everyone who could follow a step-by-step argument would be Gauss..."

Scott Aaronson, MIT

NP-Hard

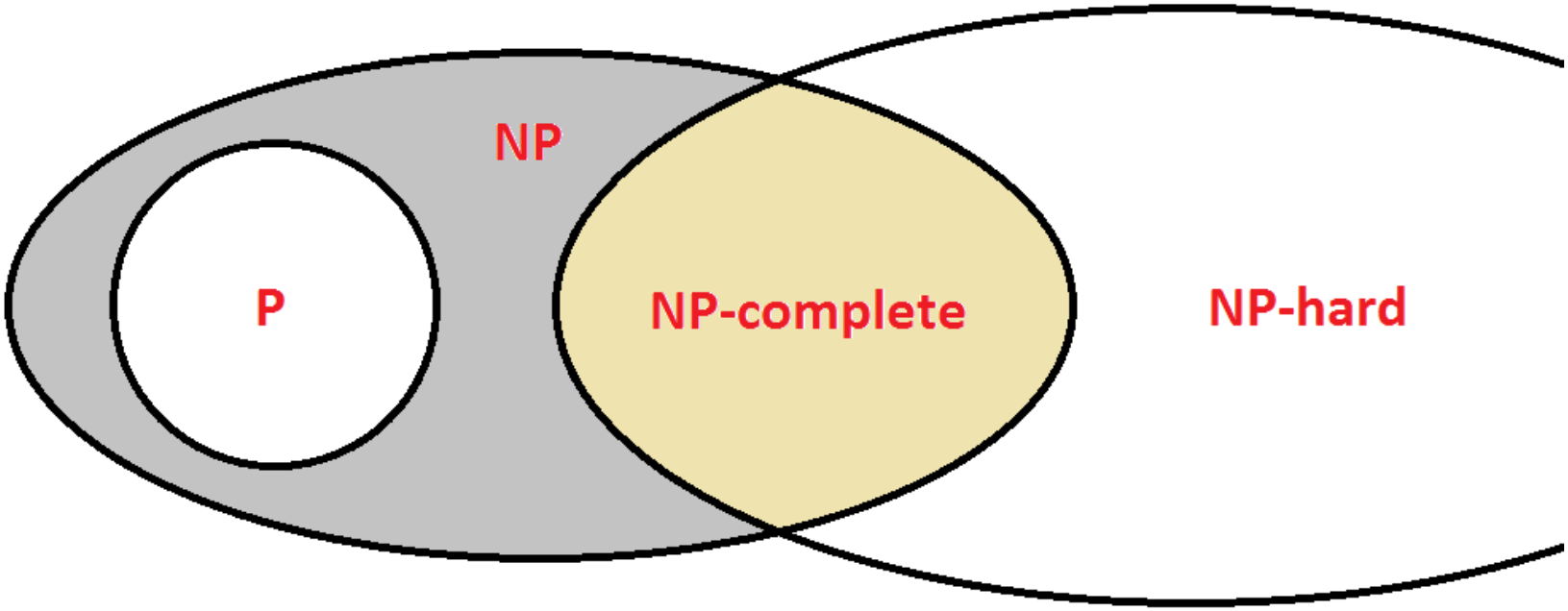


NP-Hard



Classe NP-Hard: É a classe dos problemas (*não só de decisão*) aos quais qualquer problema de NP pode ser reduzido em tempo polinomial.

NP-Hard



Classe NP-Hard: É a classe dos problemas (*não só de decisão*) aos quais qualquer problema de NP pode ser reduzido em tempo polinomial.

Classe NP-Complete: NP-complete = NP-hard $\cap$ NP.

NP-Hard

Problemas NP-complete e NP-hard podem ser considerados como mais difíceis que todos os problemas NP, porque qualquer problema NP pode ser reduzido a eles (em tempo polinomial).

3-SAT: Exemplo

12 amigos vão montar uma pizza. Cada um deve fazer uma lista com 3 alternativas a respeito dos seguintes ingredientes:

3-SAT: Exemplo

12 amigos vão montar uma pizza. Cada um deve fazer uma lista com 3 alternativas a respeito dos seguintes ingredientes:

Tomate Frango Catupiry

3-SAT: Exemplo

12 amigos vão montar uma pizza. Cada um deve fazer uma lista com 3 alternativas a respeito dos seguintes ingredientes:

Tomate Frango Catupiry

O problema é decidir se existe uma pizza que agrada todo mundo.

3-SAT: Exemplo

Arnaldo	sem tomate	**OU**	com frango	**OU**	com catupiry
Bernaldo	sem tomate	**OU**	com frango	**OU**	sem catupiry
Cernaldo	sem tomate	**OU**	sem frango	**OU**	com catupiry
Dernaldo	sem tomate	**OU**	sem frango	**OU**	sem catupiry
Ernaldo	sem frango	**OU**	com tomate	**OU**	com catupiry
Fernaldo	sem frango	**OU**	com tomate	**OU**	sem catupiry
Gernaldo	sem frango	**OU**	sem tomate	**OU**	com catupiry
Hernaldo	sem frango	**OU**	sem tomate	**OU**	sem catupiry
Irnaldo	sem catupiry	**OU**	com tomate	**OU**	com frango
Jernaldo	sem catupiry	**OU**	com tomate	**OU**	sem frango
Kernaldo	sem catupiry	**OU**	sem tomate	**OU**	com frango
Lernaldo	sem catupiry	**OU**	sem tomate	**OU**	sem frango

3-SAT: Exemplo

3-SAT

O 3-SAT é o problema de tentar satisfazer a vontade de todos, sendo que cada um oferece 3 alternativas, e fica feliz com qualquer uma delas.

3-SAT

O 3-SAT é o problema de tentar satisfazer a vontade de todos, sendo que cada um oferece 3 alternativas, e fica feliz com qualquer uma delas. SAT vem de *satisfiability* (factibilidade).

3-SAT

O 3-SAT é o problema de tentar satisfazer a vontade de todos, sendo que cada um oferece 3 alternativas, e fica feliz com qualquer uma delas. SAT vem de *satisfiability* (factibilidade).

Satisfazibilidade?

3-SAT

O 3-SAT é o problema de tentar satisfazer a vontade de todos, sendo que cada um oferece 3 alternativas, e fica feliz com qualquer uma delas. SAT vem de *satisfiability* (factibilidade).

Satisfazibilidade?

Não existem restrições práticas sobre as alternativas de cada um:

3-SAT

O 3-SAT é o problema de tentar satisfazer a vontade de todos, sendo que cada um oferece 3 alternativas, e fica feliz com qualquer uma delas. SAT vem de *satisfiability* (factibilidade).

Satisfazibilidade?

Não existem restrições práticas sobre as alternativas de cada um:

Topa tudo: com presunto **OU** com milho **OU** sem presunto

3-SAT

O 3-SAT é o problema de tentar satisfazer a vontade de todos, sendo que cada um oferece 3 alternativas, e fica feliz com qualquer uma delas. SAT vem de *satisfiability* (factibilidade).

Satisfazibilidade?

Não existem restrições práticas sobre as alternativas de cada um:

Topa tudo: com presunto **OU** com milho **OU** sem presunto
Cara chato: com atum **OU** com atum **OU** com atum

3-SAT

O 3-SAT é o problema de tentar satisfazer a vontade de todos, sendo que cada um oferece 3 alternativas, e fica feliz com qualquer uma delas. SAT vem de *satisfiability* (factibilidade).

Satisfazibilidade?

Não existem restrições práticas sobre as alternativas de cada um:

Topa tudo: com presunto **OU** com milho **OU** sem presunto
Cara chato: com atum **OU** com atum **OU** com atum
Cara não tão chato: sem atum **OU** sem atum **OU** sem atum

3-SAT

3-SAT

É o problema de decidir se uma fórmula Booleana no formato

$$(x_1 \vee y_1 \vee z_1) \wedge (x_2 \vee y_2 \vee z_2) \wedge \cdots \wedge (x_N \vee y_N \vee z_N)$$

é factível.

3-SAT

3-SAT

É o problema de decidir se uma fórmula Booleana no formato

$$(x_1 \vee y_1 \vee z_1) \wedge (x_2 \vee y_2 \vee z_2) \wedge \cdots \wedge (x_N \vee y_N \vee z_N)$$

é factível.

$\vee$: ou (OR)

$\wedge$: e (AND)

3-SAT

3-SAT

É o problema de decidir se uma fórmula Booleana no formato

$$(x_1 \vee y_1 \vee z_1) \wedge (x_2 \vee y_2 \vee z_2) \wedge \cdots \wedge (x_N \vee y_N \vee z_N)$$

é factível.

$\vee$: ou (OR)
x_i, y_i, z_i são as *literais*

$\wedge$: e (AND)
$(x_i \vee y_i \vee z_i)$ são as *cláusulas*

3-SAT

Exemplo: $(\neg a_1 \vee a_2 \vee \neg a_3) \wedge (\neg a_2 \vee \neg a_3 \vee a_4) \wedge (a_1 \vee \neg a_2 \vee \neg a_4)$

3-SAT

Exemplo: $(\neg a_1 \vee a_2 \vee \neg a_3) \wedge (\neg a_2 \vee \neg a_3 \vee a_4) \wedge (a_1 \vee \neg a_2 \vee \neg a_4)$

- Existem variáveis $a_1, a_2, a_3, \ldots$

3-SAT

Exemplo: $(\neg a_1 \vee a_2 \vee \neg a_3) \wedge (\neg a_2 \vee \neg a_3 \vee a_4) \wedge (a_1 \vee \neg a_2 \vee \neg a_4)$

- Existem variáveis $a_1, a_2, a_3, \ldots$
- Cada variável assume um valor: positivo (YES) ou negativo (NO)

3-SAT

Exemplo: $(\neg a_1 \vee a_2 \vee \neg a_3) \wedge (\neg a_2 \vee \neg a_3 \vee a_4) \wedge (a_1 \vee \neg a_2 \vee \neg a_4)$

- Existem variáveis $a_1, a_2, a_3, \ldots$
- Cada variável assume um valor: positivo (YES) ou negativo (NO)
- As literais ocorrem como *literais positivas* $(x_i = a_j)$ ou *literais negativas* $(x_i = \neg a_j)$

3-SAT

Exemplo: $(\neg a_1 \vee a_2 \vee \neg a_3) \wedge (\neg a_2 \vee \neg a_3 \vee a_4) \wedge (a_1 \vee \neg a_2 \vee \neg a_4)$

- Existem variáveis $a_1, a_2, a_3, \ldots$
- Cada variável assume um valor: positivo (YES) ou negativo (NO)
- As literais ocorrem como *literais positivas* ($x_i = a_j$) ou *literais negativas* ($x_i = \neg a_j$)
- O valor de cada literal é determinado pelo valor da variável

3-SAT

Exemplo: $(\neg a_1 \vee a_2 \vee \neg a_3) \wedge (\neg a_2 \vee \neg a_3 \vee a_4) \wedge (a_1 \vee \neg a_2 \vee \neg a_4)$

- Existem variáveis $a_1, a_2, a_3, \ldots$
- Cada variável assume um valor: positivo (YES) ou negativo (NO)
- As literais ocorrem como *literais positivas* $(x_i = a_j)$ ou *literais negativas* $(x_i = \neg a_j)$
- O valor de cada literal é determinado pelo valor da variável
- O valor de cada cláusula $C_i = (x_i \vee y_i \vee z_i)$ é YES se pelo menos uma das 3 literais é YES, caso contrário é NO

3-SAT

Exemplo: $(\neg a_1 \vee a_2 \vee \neg a_3) \wedge (\neg a_2 \vee \neg a_3 \vee a_4) \wedge (a_1 \vee \neg a_2 \vee \neg a_4)$

- Existem variáveis $a_1, a_2, a_3, \ldots$
- Cada variável assume um valor: positivo (YES) ou negativo (NO)
- As literais ocorrem como *literais positivas* $(x_i = a_j)$ ou *literais negativas* $(x_i = \neg a_j)$
- O valor de cada literal é determinado pelo valor da variável
- O valor de cada cláusula $C_i = (x_i \vee y_i \vee z_i)$ é YES se pelo menos uma das 3 literais é YES, caso contrário é NO
- O valor da expressão $C_1 \wedge C_2 \wedge \ldots \wedge C_N$ é YES se todas as cláusulas são YES, caso contrário é NO

3-SAT

Exemplo: $B = (a \vee b \vee \neg c) \wedge (\neg a \vee \neg b \vee c)$

3-SAT

Exemplo: $B = (a \vee b \vee \neg c) \wedge (\neg a \vee \neg b \vee c)$

a	b	c	$a \vee b \vee \neg c$ $\quad \neg a \vee \neg b \vee c$	B
YES	YES	YES		
YES	YES	NO		
YES	NO	YES		
YES	NO	NO		
NO	YES	YES		
NO	YES	NO		
NO	NO	YES		
NO	NO	NO		

3-SAT

Exemplo: $B = (a \vee b \vee \neg c) \wedge (\neg a \vee \neg b \vee c)$

a	b	c	$a \vee b \vee \neg c$	$\neg a \vee \neg b \vee c$	B
YES	YES	YES	YES	YES	YES
YES	YES	NO	YES	NO	NO
YES	NO	YES	YES	YES	YES
YES	NO	NO	YES	YES	YES
NO	YES	YES	YES	YES	YES
NO	YES	NO	YES	YES	YES
NO	NO	YES	NO	YES	NO
NO	NO	NO	YES	YES	YES

3-SAT e a classe NP-hard

3-SAT é um problema de decisão. O tamanho da entrada pode ser o número de literais, variáveis, ou cláusulas.

3-SAT e a classe NP-hard

3-SAT é um problema de decisão. O tamanho da entrada pode ser o número de literais, variáveis, ou cláusulas. 3-SAT é NP-complete (Stephen Cook, 1971).

3-SAT e a classe NP-hard

3-SAT é um problema de decisão. O tamanho da entrada pode ser o número de literais, variáveis, ou cláusulas. 3-SAT é NP-complete (Stephen Cook, 1971). Também virou um ponto de partida para demonstrar que outros problemas são NP-hard ou NP-complete.

3-SAT e a classe NP-hard

3-SAT é um problema de decisão. O tamanho da entrada pode ser o número de literais, variáveis, ou cláusulas. 3-SAT é NP-complete (Stephen Cook, 1971). Também virou um ponto de partida para demonstrar que outros problemas são NP-hard ou NP-complete.

Seja P_j um problema de decisão de tamanho j.

3-SAT e a classe NP-hard

3-SAT é um problema de decisão. O tamanho da entrada pode ser o número de literais, variáveis, ou cláusulas. 3-SAT é NP-complete (Stephen Cook, 1971). Também virou um ponto de partida para demonstrar que outros problemas são NP-hard ou NP-complete.

Seja P_j um problema de decisão de tamanho j. Suponha que existe um polinômio $p(x)$ tal que, para qualquer fórmula booleana B_n de tamanho n (em formato 3-SAT), resolver $P_{p(n)}$ é o mesmo que resolver B_n.

3-SAT e a classe NP-hard

3-SAT é um problema de decisão. O tamanho da entrada pode ser o número de literais, variáveis, ou cláusulas. 3-SAT é NP-complete (Stephen Cook, 1971). Também virou um ponto de partida para demonstrar que outros problemas são NP-hard ou NP-complete.

Seja P_j um problema de decisão de tamanho j. Suponha que existe um polinômio $p(x)$ tal que, para qualquer fórmula booleana B_n de tamanho n (em formato 3-SAT), resolver $P_{p(n)}$ é o mesmo que resolver B_n.

Então P é um problema NP-hard:

3-SAT e a classe NP-hard

3-SAT é um problema de decisão. O tamanho da entrada pode ser o número de literais, variáveis, ou cláusulas. 3-SAT é NP-complete (Stephen Cook, 1971). Também virou um ponto de partida para demonstrar que outros problemas são NP-hard ou NP-complete.

Seja P_j um problema de decisão de tamanho j. Suponha que existe um polinômio $p(x)$ tal que, para qualquer fórmula booleana B_n de tamanho n (em formato 3-SAT), resolver $P_{p(n)}$ é o mesmo que resolver B_n.

Então P é um problema NP-hard: se você consegue resolver qualquer P_j em tempo polinomial $q(j)$,

3-SAT e a classe NP-hard

3-SAT é um problema de decisão. O tamanho da entrada pode ser o número de literais, variáveis, ou cláusulas. 3-SAT é NP-complete (Stephen Cook, 1971). Também virou um ponto de partida para demonstrar que outros problemas são NP-hard ou NP-complete.

Seja P_j um problema de decisão de tamanho j. Suponha que existe um polinômio $p(x)$ tal que, para qualquer fórmula booleana B_n de tamanho n (em formato 3-SAT), resolver $P_{p(n)}$ é o mesmo que resolver B_n.

Então P é um problema NP-hard: se você consegue resolver qualquer P_j em tempo polinomial $q(j)$, você consegue resolver qualquer fórmula booleana 3-SAT B_n,

3-SAT e a classe NP-hard

3-SAT é um problema de decisão. O tamanho da entrada pode ser o número de literais, variáveis, ou cláusulas. 3-SAT é NP-complete (Stephen Cook, 1971). Também virou um ponto de partida para demonstrar que outros problemas são NP-hard ou NP-complete.

Seja P_j um problema de decisão de tamanho j. Suponha que existe um polinômio $p(x)$ tal que, para qualquer fórmula booleana B_n de tamanho n (em formato 3-SAT), resolver $P_{p(n)}$ é o mesmo que resolver B_n.

Então P é um problema NP-hard: se você consegue resolver qualquer P_j em tempo polinomial $q(j)$, você consegue resolver qualquer fórmula booleana 3-SAT B_n, em tempo polinomial $q(p(n))$.

3-SAT e a classe NP

3-SAT talvez não seja polinomial. Uma conjectura diz que é na melhor das hipóteses exponencial.

3-SAT e a classe NP

3-SAT talvez não seja polinomial. Uma conjectura diz que é na melhor das hipóteses exponencial. Mas certamente está em NP.

3-SAT e a classe NP

3-SAT talvez não seja polinomial. Uma conjectura diz que é na melhor das hipóteses exponencial. Mas certamente está em NP.

Algoritmo: Testar cada possível valor (YES ou NO) de cada variável.

3-SAT e a classe NP

3-SAT talvez não seja polinomial. Uma conjectura diz que é na melhor das hipóteses exponencial. Mas certamente está em NP.

Algoritmo: Testar cada possível valor (YES ou NO) de cada variável.

Se são M variáveis, são 2^M casos para testar.

It's me, Mario!

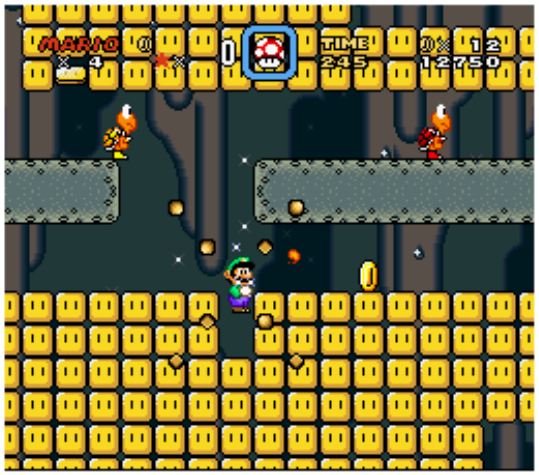
Problema

Uma dada fase do Mario é possível?

It's me, Mario!


MARIO
× 5
TIME
227
7800


MARIO
×20
73
TIME
286
214810


MARIO
TIME
245
12750

It's me, Mario!

It's me, Mario!

It's me, Mario!

It's me, Mario!

It's me, Mario!

Objetivo: Provar que o problema de decisão

Dada uma fase qualquer do Mario, é possível passar dela?

é NP-hard

It's me, Mario!

Objetivo: Provar que o problema de decisão

Dada uma fase qualquer do Mario, é possível passar dela?

é NP-hard *com relação ao tamanho da fase*.

It's me, Mario!

Objetivo: Provar que o problema de decisão

Dada uma fase qualquer do Mario, é possível passar dela?

é NP-hard *com relação ao tamanho da fase*.

Observação: Isso vai implicar no seguinte:

Se existir uma maneira de dizer em tempo polinomial se uma dada fase do Mario é possível, então

It's me, Mario!

Objetivo: Provar que o problema de decisão

Dada uma fase qualquer do Mario, é possível passar dela?

é NP-hard *com relação ao tamanho da fase*.

Observação: Isso vai implicar no seguinte:

Se existir uma maneira de dizer em tempo polinomial se uma dada fase do Mario é possível, então P=NP!

It's me, Mario!

Objetivo: Provar que o problema de decisão

Dada uma fase qualquer do Mario, é possível passar dela?

é NP-hard *com relação ao tamanho da fase*.

Observação: Isso vai implicar no seguinte:

Se existir uma maneira de dizer em tempo polinomial se uma dada fase do Mario é possível, então P=NP!

NP?

Os autores fazem questão de lembrar que, considerando todos os jogos mencionados, é complicado analisar se este problema está em NP.

NP?

Os autores fazem questão de lembrar que, considerando todos os jogos mencionados, é complicado analisar se este problema está em NP. Mas:

O problema para Super Mario Bros. está em NP.

NP?

Os autores fazem questão de lembrar que, considerando todos os jogos mencionados, é complicado analisar se este problema está em NP. Mas:

O problema para Super Mario Bros. está em NP.

Basicamente porque as únicas coisas que se mexem são os koopas (e goombas e outros inimigos) e os cascos,

NP?

Os autores fazem questão de lembrar que, considerando todos os jogos mencionados, é complicado analisar se este problema está em NP. Mas:

O problema para Super Mario Bros. está em NP.

Basicamente porque as únicas coisas que se mexem são os koopas (e goombas e outros inimigos) e os cascos, mas o Mario não pode carregar o casco.

3-SAT na fase do Mario

Abordagem: Dada uma fórmula 3-SAT, construir uma fase do Mario que só pode ser vencida se a fórmula for factível.

3-SAT na fase do Mario

Abordagem: Dada uma fórmula 3-SAT, construir uma fase do Mario que só pode ser vencida se a fórmula for factível.

Suponha que a fórmula é

$$(a \vee \neg b \vee c) \wedge (a \vee \neg b \vee \neg c)$$

3-SAT na fase do Mario

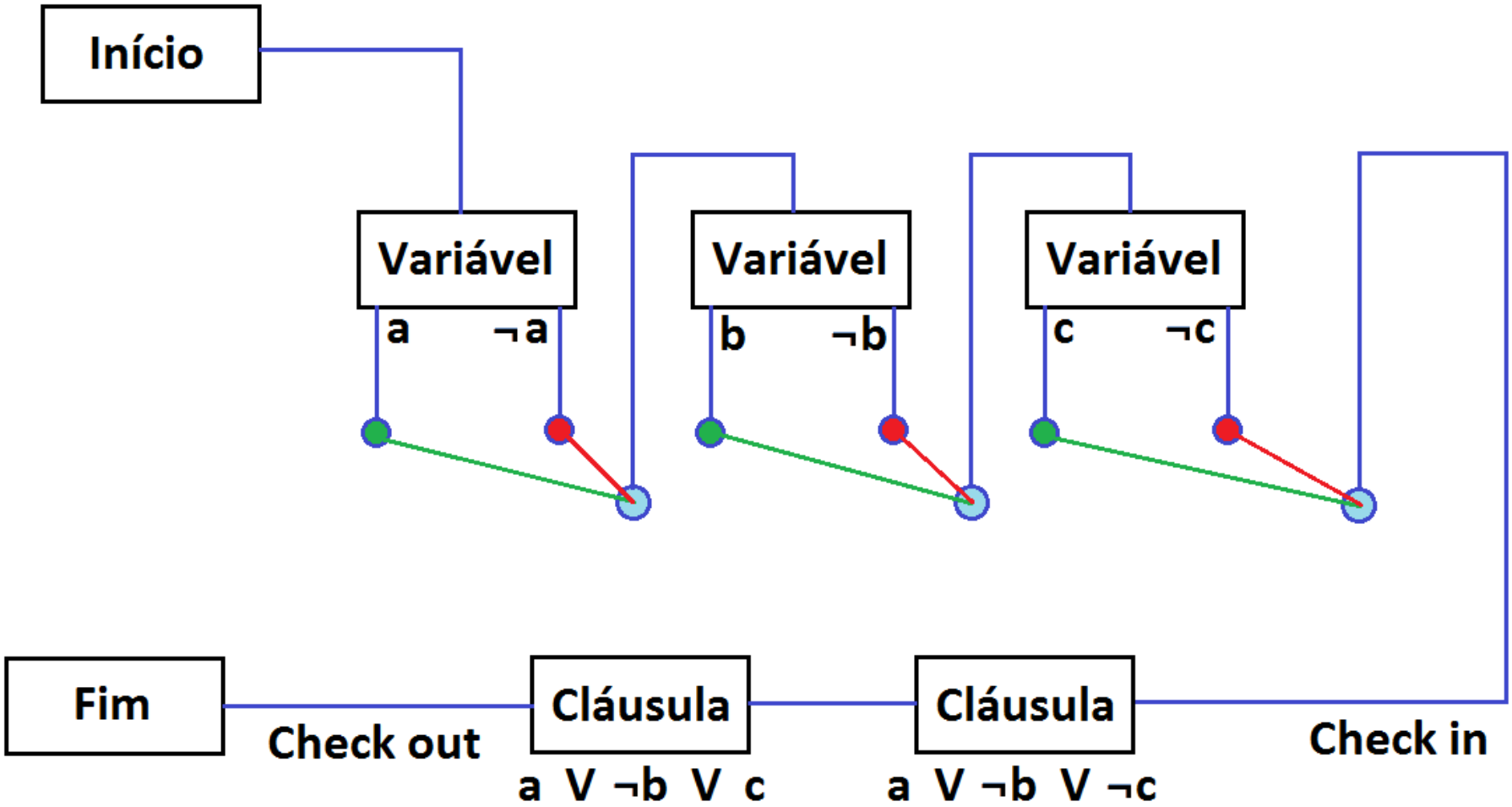
Abordagem: Dada uma fórmula 3-SAT, construir uma fase do Mario que só pode ser vencida se a fórmula for factível.

Suponha que a fórmula é

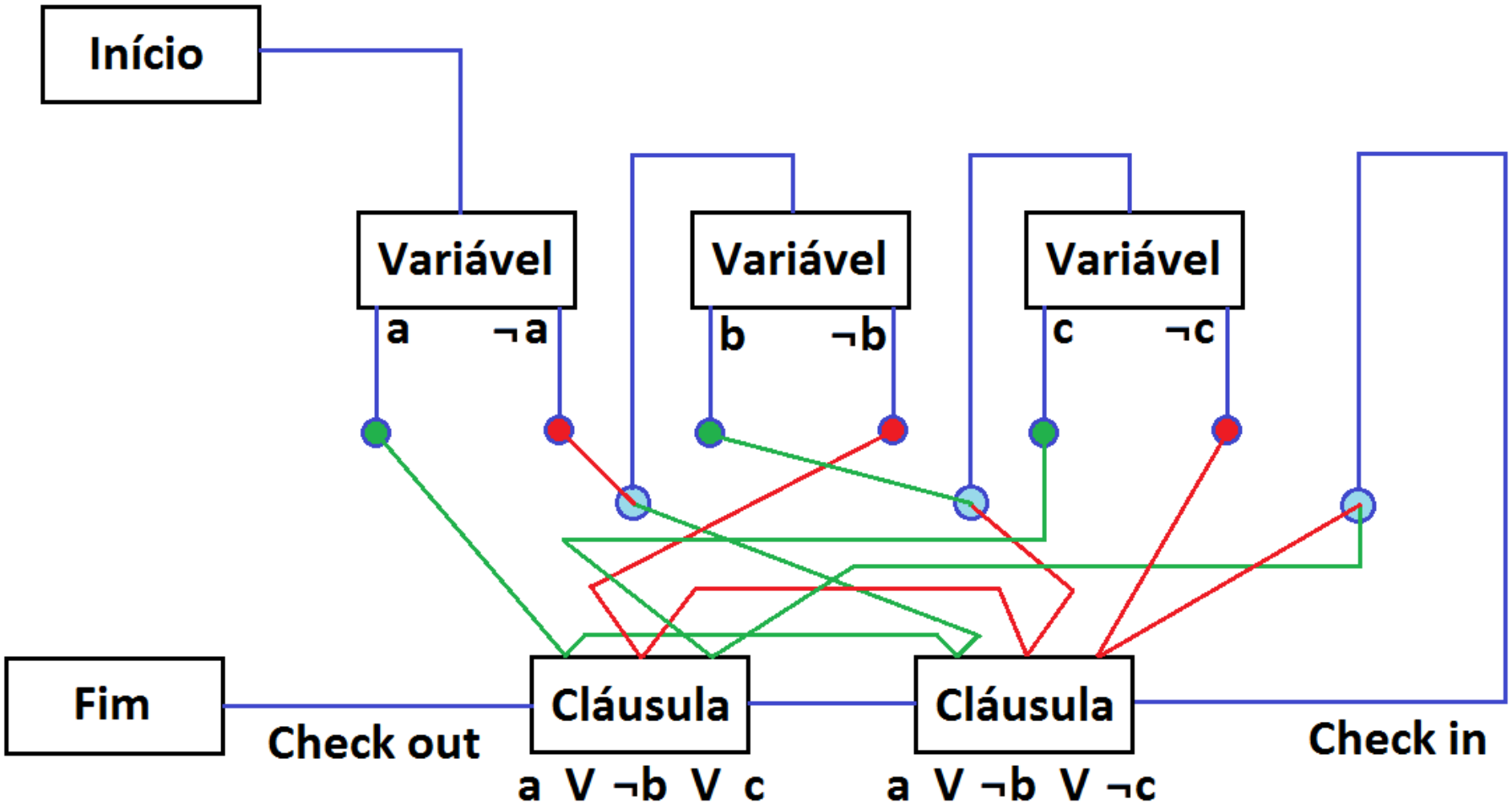
$$(a \vee \neg b \vee c) \wedge (a \vee \neg b \vee \neg c)$$

O layout da fase é o seguinte:

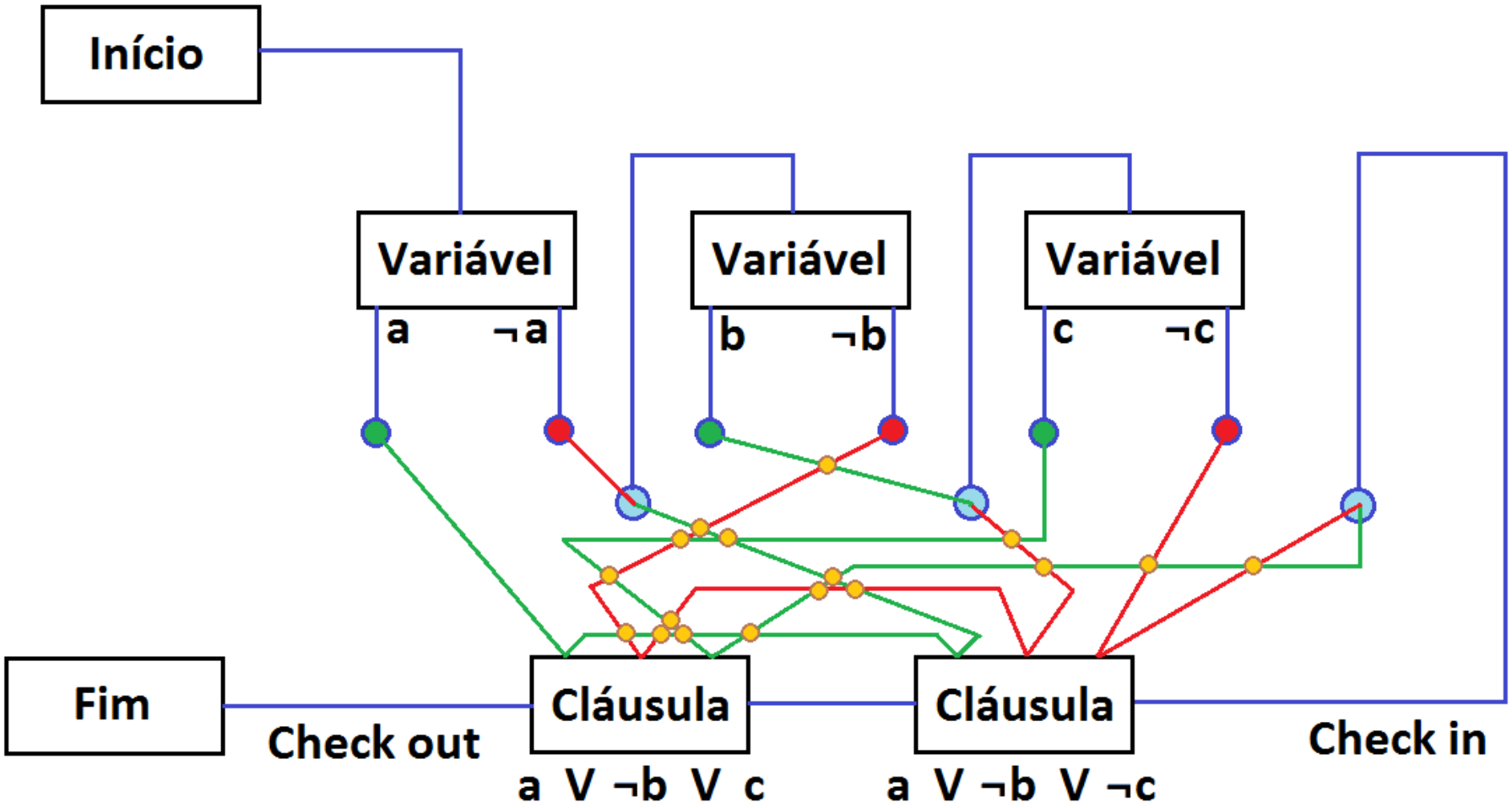
Fase


Início
Variável
a
¬a
Variável
b
¬b
Variável
c
¬c
Fim
Check out
Cláusula
a V ¬b V c
Cláusula
a V ¬b V ¬c
Check in

Fase


Início
Variável
a
¬a
Variável
b
¬b
Variável
c
¬c
Fim
Check out
Cláusula
a V ¬b V c
Cláusula
a V ¬b V ¬c
Check in

Fase


Início
Variável
a
¬a
Variável
b
¬b
Variável
c
¬c
Fim
Check out
Cláusula
a V ¬b V c
Cláusula
a V ¬b V ¬c
Check in

Mecanismos

Está claro que o tamanho da fase aumenta polinomialmente em relação ao tamanho da fórmula.

Mecanismos

Está claro que o tamanho da fase aumenta polinomialmente em relação ao tamanho da fórmula. (*Está?*)

Mecanismos

Está claro que o tamanho da fase aumenta polinomialmente em relação ao tamanho da fórmula. (*Está?*)

Só precisamos implementar os mecanismos:

- Início
- Variável
- Cláusula
- Cruzamento
- Fim

Mecanismos

Os mecanismos de Início e Fim são exatamente o que você pensa.

Mecanismos

Os mecanismos de Início e Fim são exatamente o que você pensa.

Mas os outros mecanismos poderiam falhar se o Mario estiver pequeno.

Mecanismos

Os mecanismos de Início e Fim são exatamente o que você pensa.

Mas os outros mecanismos poderiam falhar se o Mario estiver pequeno. A solução é exigir que o Mario esteja grande ao final para poder ganhar da fase.

Mecanismo de Início

Mecanismo de Início

Mecanismo de Fim

Mecanismo de Variável

Propriedades do mecanismo de Variável:

Mecanismo de Variável

Propriedades do mecanismo de Variável:

- Entrada por cima

Mecanismo de Variável

Propriedades do mecanismo de Variável:

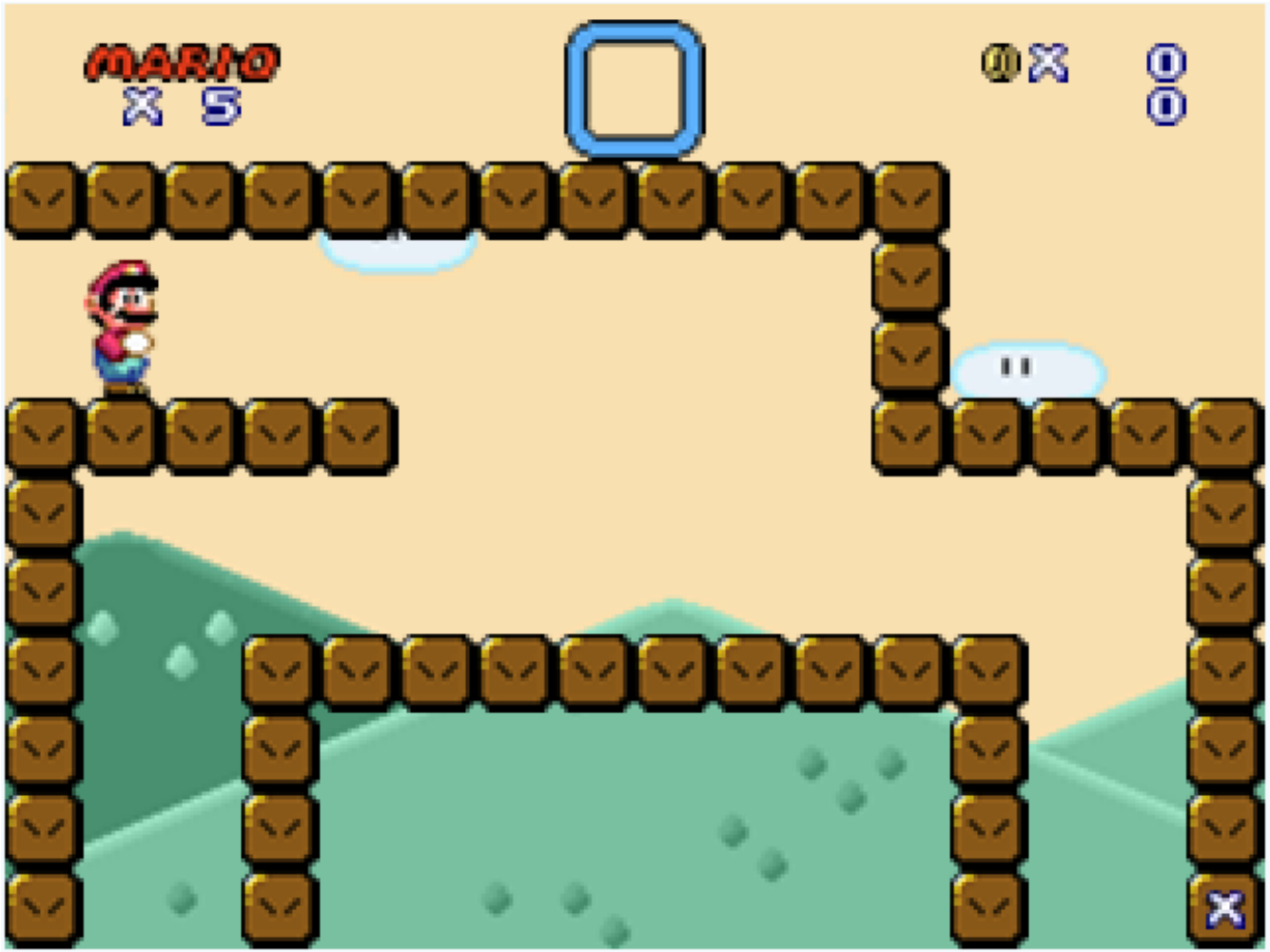
- Entrada por cima
- Duas saídas por baixo, Mario só pode escolher uma

Mecanismo de Variável

Propriedades do mecanismo de Variável:

- Entrada por cima
- Duas saídas por baixo, Mario só pode escolher uma
- Não é possível retornar

Mecanismo de Variável



Mecanismo de Cláusula

Propriedades do mecanismo de Cláusula:

Mecanismo de Cláusula

Propriedades do mecanismo de Cláusula:

- Só pode ser *atravessado* da direita para a esquerda

Mecanismo de Cláusula

Propriedades do mecanismo de Cláusula:

- Só pode ser *atravessado* da direita para a esquerda
- Inicialmente trancado

Mecanismo de Cláusula

Propriedades do mecanismo de Cláusula:

- Só pode ser *atravessado* da direita para a esquerda
- Inicialmente trancado
- Só pode ser atravessado se estiver destrancado

Mecanismo de Cláusula

Propriedades do mecanismo de Cláusula:

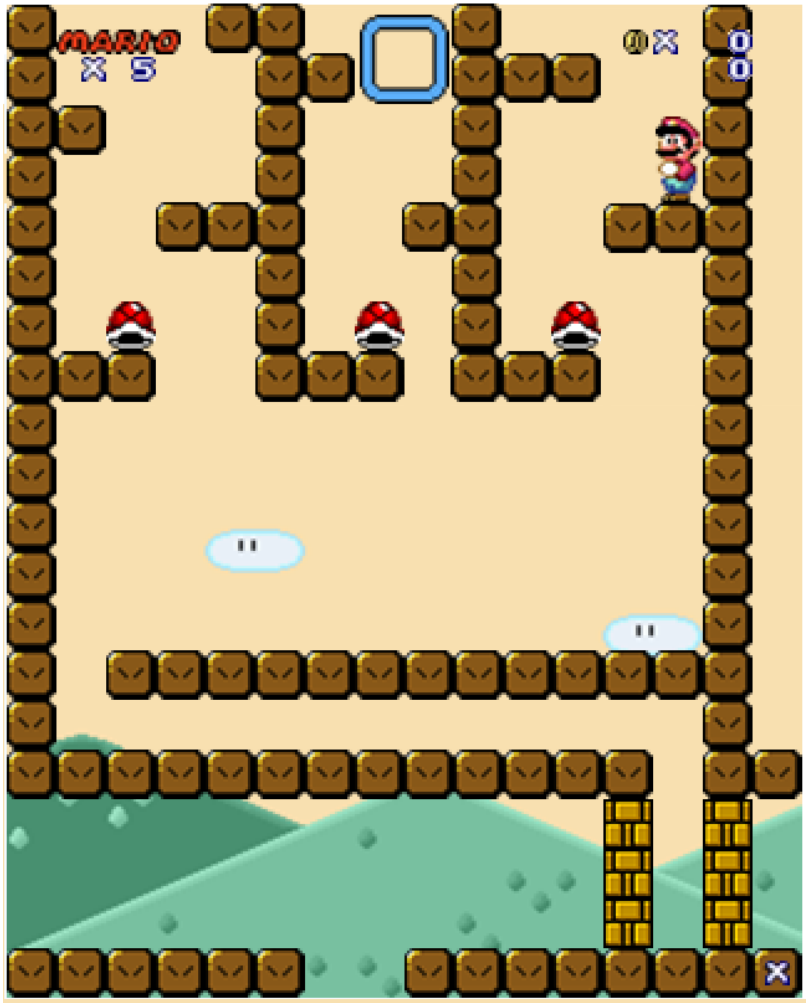
- Só pode ser *atravessado* da direita para a esquerda
- Inicialmente trancado
- Só pode ser atravessado se estiver destrancado
- Pode ser apenas *visitado* de cima

Mecanismo de Cláusula

Propriedades do mecanismo de Cláusula:

- Só pode ser *atravessado* da direita para a esquerda
- Inicialmente trancado
- Só pode ser atravessado se estiver destrancado
- Pode ser apenas *visitado* de cima
- Só pode ser destrancado de cima, através de qualquer uma de três entradas

Mecanismo de Cláusula



Mecanismo de Cruzamento

Propriedades do mecanismo de Cruzamento:

Mecanismo de Cruzamento

Propriedades do mecanismo de Cruzamento:

- Pode ser cruzado na vertical ou na horizontal

Mecanismo de Cruzamento

Propriedades do mecanismo de Cruzamento:

- Pode ser cruzado na vertical ou na horizontal
- Não existe vazamento

Mecanismo de Cruzamento

Propriedades do mecanismo de Cruzamento:

- Pode ser cruzado na vertical ou na horizontal
- Não existe vazamento
- Pode ser unidirecional

Mecanismo de Cruzamento

Mecanismo de Cruzamento

Se o Mario pode carregar cascos, a alternativa é:

Mecanismo de Cruzamento

Se o Mario pode carregar cascos, a alternativa é:

Cruzamento unidirecional?

Se precisar ser atravessado da direita pra esquerda, tudo bem.

Cruzamento unidirecional?

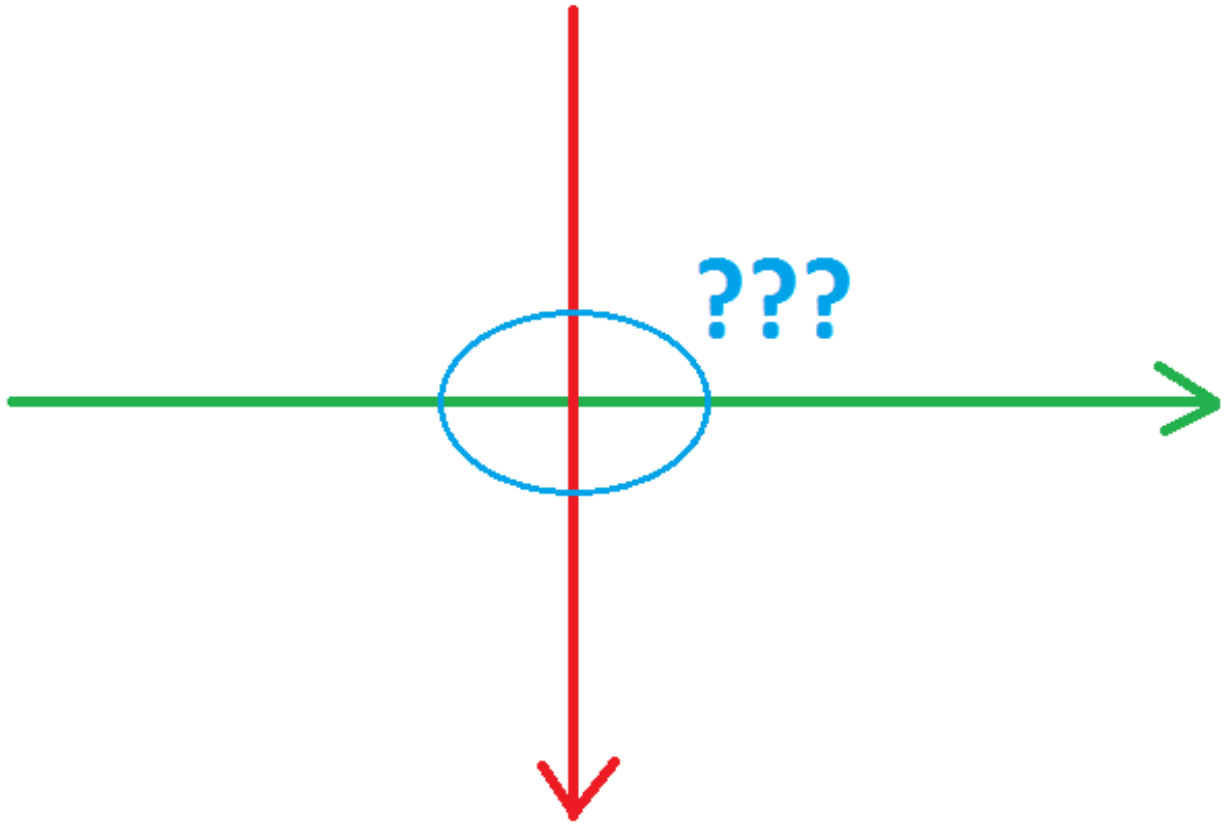
Se precisar ser atravessado da direita pra esquerda, tudo bem.

Mecanismo de Cruzamento

Mas e se precisar ser atravessado de cima pra baixo?

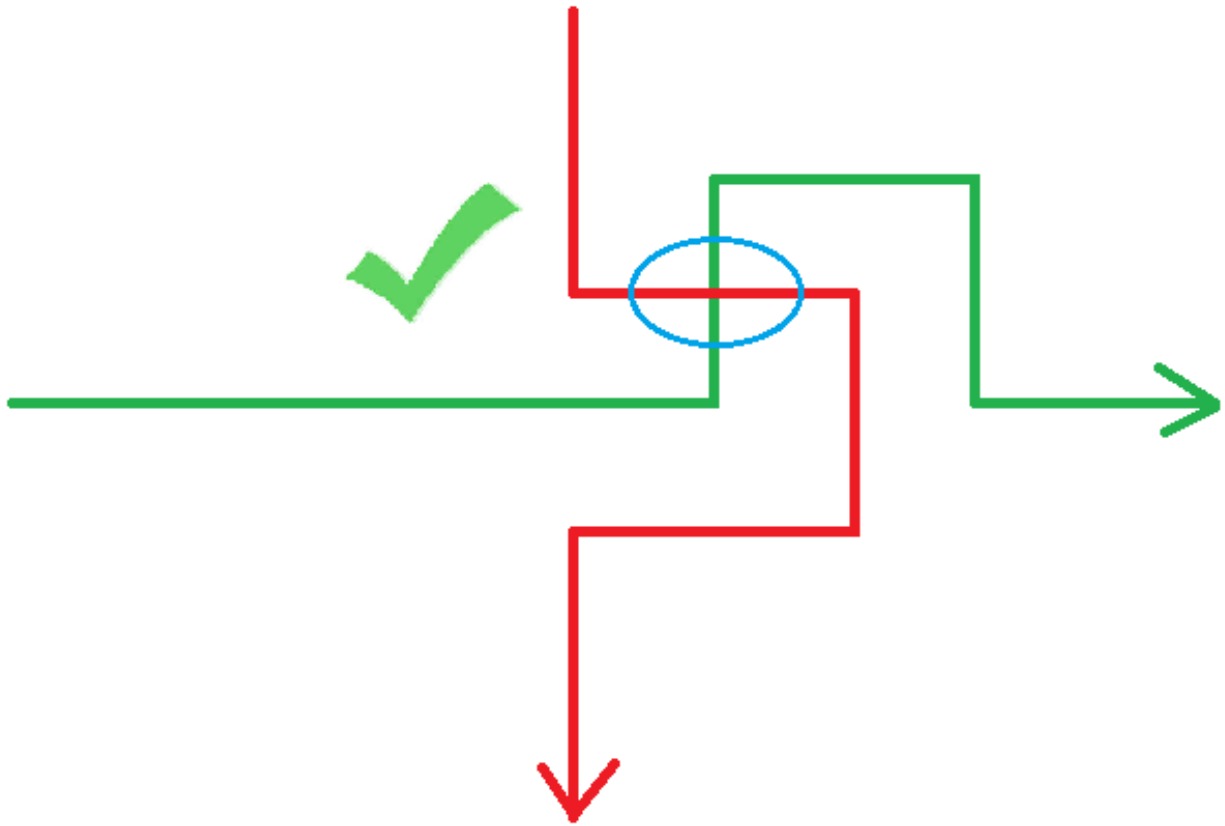
Mecanismo de Cruzamento

Mas e se precisar ser atravessado de cima pra baixo?



Mecanismo de Cruzamento

Solução:



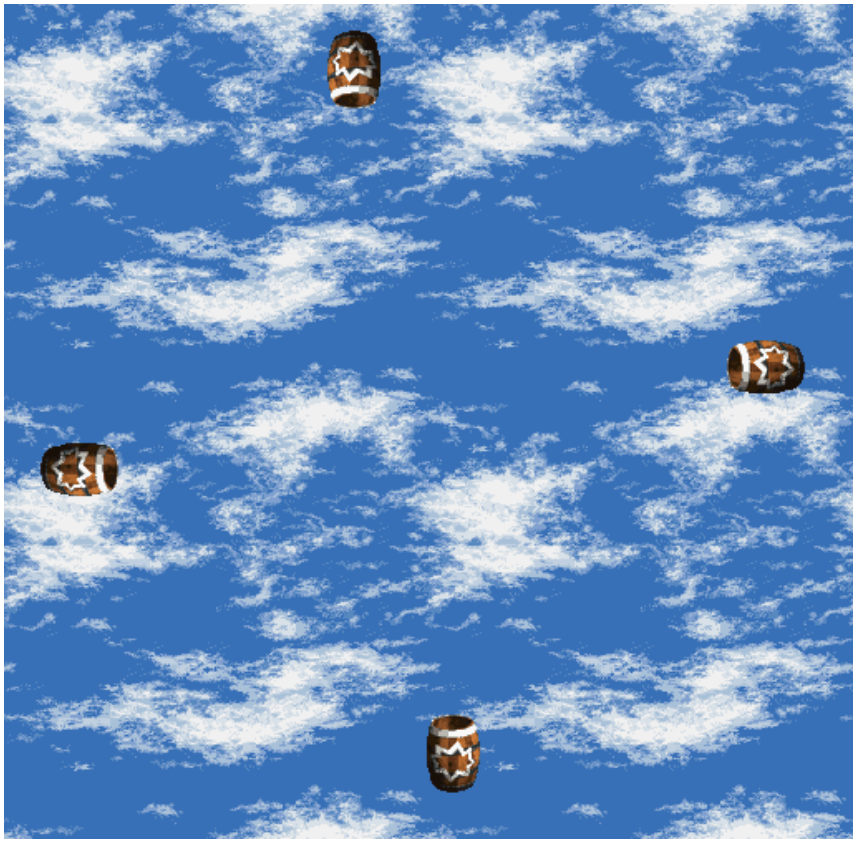
cqd

Donkey Kong Country

Cruzamento no Donkey Kong:

Donkey Kong Country

Cruzamento no Donkey Kong:



Pokémon

A abordagem é semelhante para o *Pokémon*, mas o layout da fase é diferente e envolve outros mecanismos:

Pokémon

A abordagem é semelhante para o *Pokémon*, mas o layout da fase é diferente e envolve outros mecanismos:

- Start
- One-way
- Fork
- H
- No-reverse
- Lock
- XOR-crossover
- End

Os detalhes estão em outro artigo, sobre o jogo *PushPush*.

Pokémon

- **Player**: Gastly nível 1

Pokémon

- **Player**: Gastly nível 1
- **Treinador tipo 1**: Electrode nível 100

Pokémon

- **Player**: Gastly nível 1
- **Treinador tipo 1**: Electrode nível 100 que só tem *Selfdestruct*

Pokémon

- **Player**: Gastly nível 1
- **Treinador tipo 1**: Electrode nível 100 que só tem *Selfdestruct*
- **Treinador tipo 2**: Alakazam nível 100

Pokémon

- **Player**: Gastly nível 1
- **Treinador tipo 1**: Electrode nível 100 que só tem *Selfdestruct*
- **Treinador tipo 2**: Alakazam nível 100 com *Psychic*

Desafio

$$a \vee b \vee \neg c$$

Referências

- Aloupis G., Demaine E., Guo A., *Classic Nintendo Games are (NP-)Hard*. arXiv:1203.1895v1. **2012**. (Acesso em 01 abril de 2013)
- Demaine E., Demaine M. L., O'Rourke J., *PushPush and Push-1 are NP-Hard in 2D*. Proceedings of the 12th Annual Canadian Conference on Computational Geometry. **2000**, pp. 211-219.
- Garey M. R., Johnson D. S., *COMPUTERS AND INTRACTABILITY: A guide to the Theory of NP-Completeness*. W. H. Freeman and Company, New York. **1978**.

Referências

- http://www.videogamesprites.net/ (Acesso em 08 de outubro de 2013)
- *Super Mario Flash 2*:

  http://www.pouetpu-games.com/index.php?section=2&game_id=2

  (Acesso em 08 de outubro de 2013 e pro resto da minha vida)
- Imagens tiradas da internet.


THE END