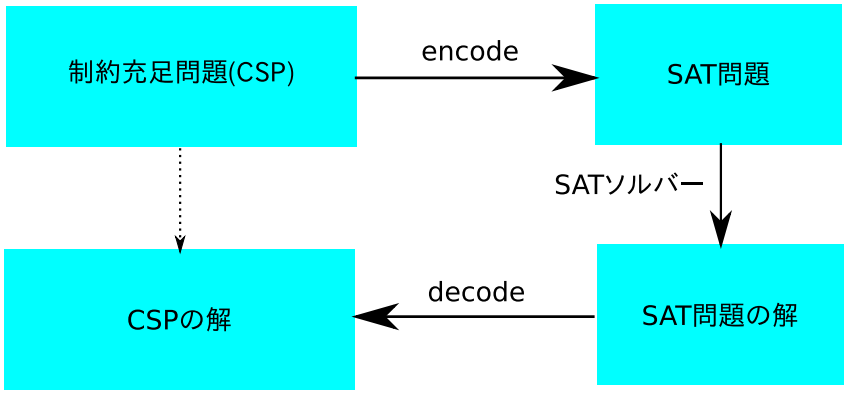
パズルを Sugar 制約ソルバーで解く

田村直之

神戸大学

第 1 回 CSPSAT 研究会
2008 年 8 月 21 日

Sugar 制約ソルバーの概要



- Sugar は**制約充足問題** (CSP) を命題論理の充足可能性判定問題 (SAT 問題) に変換 (encode) し，SAT ソルバーを用いて解を探索するシステムである．
- SAT 変換としては order encoding 法を用いている．多くの問題に対して，従来広く用いられている direct encoding や support encoding よりも高速な求解が可能である．
- 現在の所，変数間の乗除算には対応していない．

http://bach.istc.kobe-u.ac.jp/sugar/ ▸ Web

Sugar 制約ソルバーの特徴

- SAT ソルバーを複数回使用することにより，制約最適化問題 (COP) や MAX-CSP 問題にも対応している．
- International CSP Solver Competition で採用されている XCSP 2.1 フォーマットのベンチマーク問題 (5000 問以上) も読込可能である．
- SAT ソルバーとしては MiniSat, PicoSAT 等が利用できる．
- Cygwin を利用すれば，Windows XP 等でも動作する．
- CSP の記述には，Lisp 風のリスト表現を用いる．

第 3 回 International CSP Solver Competition ▸ Web

Sugar を Windows 上で Cygwin を用いて動かす方法 ▸ Web

Sugar の構文 (1) ▸ Web

- 変数として，ブール変数と整数変数 (有限領域) が利用できる．

変数宣言の例

```
(bool p)          ; ブール変数 p の宣言
(int x 1 4)       ; 整数変数 x ∈ {1,2,3,4} の宣言
(int y (0 2 5))   ; 整数変数 y ∈ {0,2,5} の宣言
```

Sugar の構文 ▸ Web

Sugar の構文 (2) ▸ Web

- **算術演算として，** abs, neg (−), add (+), sub (−), mul (∗), div (/), mod (%), min, max, if **が利用できる．**
- **比較演算として，** eq (=), ne (!=), le (<=), lt (<), ge (>=), gt (>) **が利用できる．**
- **論理演算として，** not (!), and (&&), or (||), imp (=>), xor, iff **が利用できる．**
- **グローバル制約として** alldifferent, weightedsum, cumulative, element **が利用できる．**

制約の例

```
(or (> x (+ y 1)) (> y x)) ; (x > y + 1) ∨ (y > x)
(alldifferent x y z)
```

Sugar **の構文** ▸ Web

CSP の例

3 × 3 魔方陣の例

```
(int x_1_1 1 9)  (int x_1_2 1 9)  (int x_1_3 1 9)
(int x_2_1 1 9)  (int x_2_2 1 9)  (int x_2_3 1 9)
(int x_3_1 1 9)  (int x_3_2 1 9)  (int x_3_3 1 9)
(alldifferent x_1_1 x_1_2 x_1_3 x_2_1 x_2_2 x_2_3 x_3_1 x_3_2 x_3_3)
(= (+ x_1_1 x_1_2 x_1_3) 15)
(= (+ x_2_1 x_2_2 x_2_3) 15)
(= (+ x_3_1 x_3_2 x_3_3) 15)
(= (+ x_1_1 x_2_1 x_3_1) 15)
(= (+ x_1_2 x_2_2 x_3_2) 15)
(= (+ x_1_3 x_2_3 x_3_3) 15)
(= (+ x_1_1 x_2_2 x_3_3) 15)
(= (+ x_1_3 x_2_2 x_3_1) 15)
```

Sugar の実行例

3 × 3 魔方陣の実行例

```
$ sugar -vv magicSquare-3x3.csp
c 0 Sugar v1-14 + minisat
c 0 .....
c 0 ENCODING magicSquare-3x3.csp TO /tmp/sugar17777.cnf
c 0 CSP : 9 integers, 0 booleans, 54 clauses, largest domain size 9
c 0 .....
c 0 SAT : 144 SAT variables, 1709 SAT clauses, 20164 bytes
c 0 .....
c 1 SOLVING /tmp/sugar17777.cnf
c 1 CMD minisat '/tmp/sugar17777.cnf' '/tmp/sugar17777.out'
c 1 .....
c 1 DECODING /tmp/sugar17777.out WITH /tmp/sugar17777.map
c 1 .....
s SATISFIABLE
a x_1_1 2
a x_1_2 9
a x_1_3 4
a x_2_1 7
a x_2_2 5
a x_2_3 3
a x_3_1 6
a x_3_2 1
a x_3_3 8
a
c 1 .....
c 1 CPU 0.75 (0.06 0.02 0.59 0.08)
c 1 END Fri Aug 15 21:56:35 2008
```

パズルを Sugar 制約ソルバーで解く (1)

パズル雑誌のニコリ等にあるパズルを Sugar で解いてみた.

なぜパズルを解いてみようと思ったのか

パズルを Sugar 制約ソルバーで解く ▸ Web

ニコリ ▸ Web

パズルを Sugar 制約ソルバーで解く (1)

パズル雑誌のニコリ等にあるパズルを Sugar で解いてみた．

なぜパズルを解いてみようと思ったのか

- **パズルが好きだから** :-)
- **でも，これまで開発した Prolog 処理系や制約ソルバーでは，なかなかニコリのパズルは解けなかった．**
- Sugar **なら解けそうだ．**

パズルを Sugar 制約ソルバーで解く ▸ Web

ニコリ ▸ Web

パズルを Sugar 制約ソルバーで解く (2)

解いてみたパズルの種類

比較的自然に制約記述できたパズル

数独
四角に切れ
カックロ
お絵かきロジック
美術館
ナンバーリンク

単一ループの条件が必要なパズル

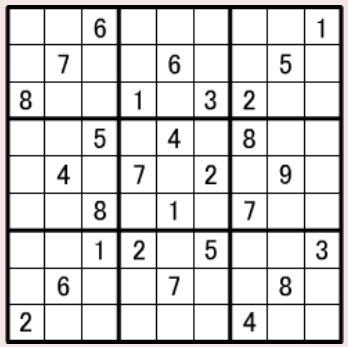
スリザーリンク
ましゅ
ヤジリン

連結条件が必要なパズル

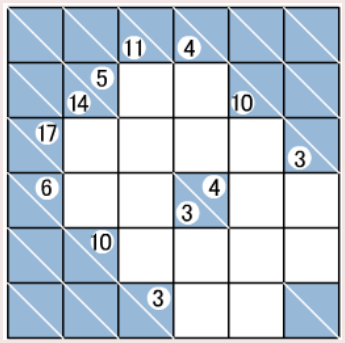
ひとりにしてくれ
フィルオミノ
橋をかけろ
へやわけ
ぬりかべ

比較的自然に制約記述できたパズル

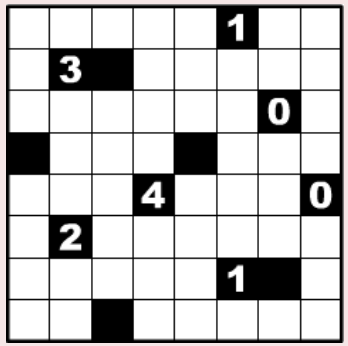
数独



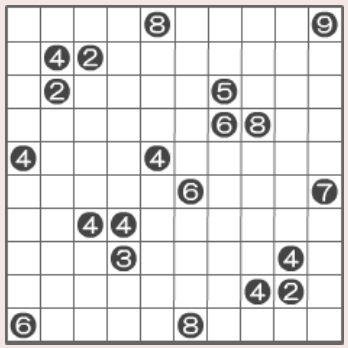
カックロ



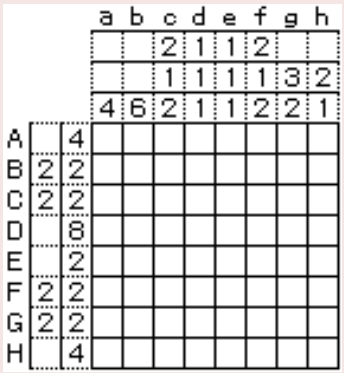
美術館



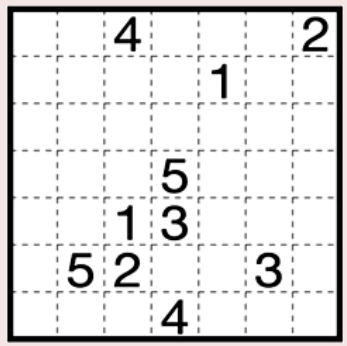
四角に切れ



お絵かきロジック



ナンバーリンク



数独

ニコリによる例題とルールの説明 ▸ Web

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

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

数独のルール

1. あいているマスに，1 から 9 までの数字のどれかを入れます．
2. タテ列（9 列あります），ヨコ列（9 列あります），太線で囲まれた 3x3 のブロック（それぞれ 9 マスあるブロックが 9 つあります）のどれにも 1 から 9 までの数字が 1 つずつ入ります．

数独を Sugar で解く

▸ Web

数独の制約記述

- *i* 行 *j* 列目のマスに整数変数 x_*i*_*j* を割り当てる．
  - 白マスの場合は，1 から 9 の値を取る変数にする．
  - 数字マスの場合は，その値を取る変数にする．
- 数字が相異なる条件を alldifferent で表す．

数独の記述例

```
(int x_0_0 1 9)   ; x_0_0 ∈ {1,2,...,9}
(int x_0_1 1 9)   ; x_0_1 ∈ {1,2,...,9}
(int x_0_2 6 6)   ; x_0_2 ∈ {6}
.....
(alldifferent x_0_0 x_0_1 ... x_0_8) ; 0 行目
.....
(alldifferent x_0_0 x_1_0 ... x_8_0) ; 0 列目
.....
```

数独を Sugar で解く

▸ Web

数独の制約記述

- *i* 行 *j* 列目のマスに整数変数 x_*i*_*j* を割り当てる．
  - 白マスの場合は，1 から 9 の値を取る変数にする．
  - 数字マスの場合は，その値を取る変数にする．
- 数字が相異なる条件を alldifferent で表す．

数独の記述例

```
(int x_0_0 1 9)   ; x_0_0 ∈ {1, 2, ..., 9}
(int x_0_1 1 9)   ; x_0_1 ∈ {1, 2, ..., 9}
(int x_0_2 6 6)   ; x_0_2 ∈ {6}
.....
(alldifferent x_0_0 x_0_1 ... x_0_8) ; 0 行目
.....
(alldifferent x_0_0 x_1_0 ... x_8_0) ; 0 列目
.....
```

数独を Sugar で解く

▸ Web

数独の制約記述

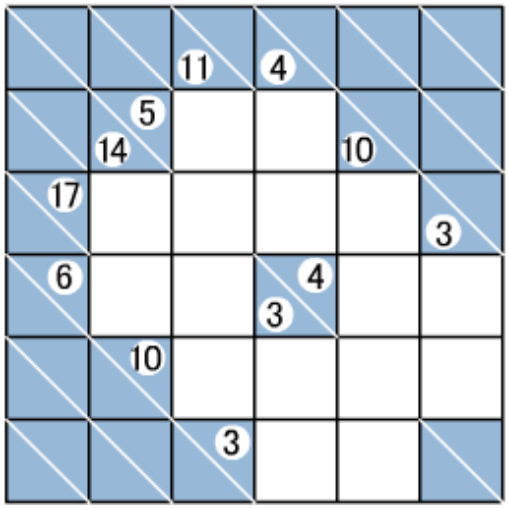
- *i* 行 *j* 列目のマスに整数変数 x_*i*_*j* を割り当てる．
  - 白マスの場合は，1 から 9 の値を取る変数にする．
  - 数字マスの場合は，その値を取る変数にする．
- **数字が相異なる条件を** alldifferent **で表す．**

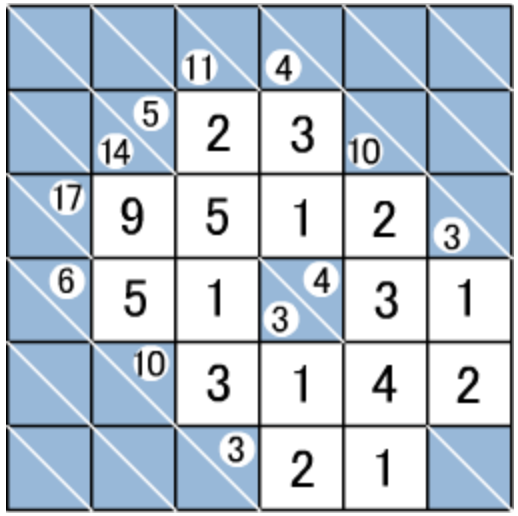
数独の記述例

```
(int x_0_0 1 9)   ; x_0_0 ∈ {1,2,...,9}
(int x_0_1 1 9)   ; x_0_1 ∈ {1,2,...,9}
(int x_0_2 6 6)   ; x_0_2 ∈ {6}
.....
(alldifferent x_0_0 x_0_1 ... x_0_8) ; 0 行目
.....
(alldifferent x_0_0 x_1_0 ... x_8_0) ; 0 列目
.....
```

カックロ

ニコリによる例題とルールの説明 ▸ Web





カックロのルール

1. あいているマスに，1 から 9 までの数字のどれかを入れます．0(ゼロ) は使いません．
2. ナナメの線の右上に出ている数字は，その右の白マスのつながりに入る数字の合計．ナナメの線の左下に出ている数字は，その下の白マスのつながりに入る数字の合計です．
3. タテヨコへの 1 つの白マスのつながりには，同じ数字は入りません．

カックロを Sugar で解く

▸ Web

カックロの制約記述

- *i* 行 *j* 列目の白マスに整数変数 x_*i*_*j* を割り当てる.
- 数字の合計の条件を記述する.
- 数字が相異なる条件を alldifferent で表す.

カックロの記述例

```
(int x_1_2 1 9)
(int x_1_3 1 9)
.....
(= (+ x_1_2 x_1_3) 5)
(alldifferent x_1_2 x_1_3)
.....
```

カックロを Sugar で解く

▸ Web

カックロの制約記述

- *i* 行 *j* 列目の白マスに整数変数 x_*i*_*j* を割り当てる.
- 数字の合計の条件を記述する.
- 数字が相異なる条件を alldifferent で表す.

カックロの記述例

```
(int x_1_2 1 9)
(int x_1_3 1 9)
.....
(= (+ x_1_2 x_1_3) 5)
(alldifferent x_1_2 x_1_3)
.....
```

カックロを Sugar で解く

▸ Web

カックロの制約記述

- *i* 行 *j* 列目の白マスに整数変数 x_*i*_*j* を割り当てる.
- **数字の合計の条件を記述する.**
- 数字が相異なる条件を alldifferent で表す.

カックロの記述例

```
(int x_1_2 1 9)
(int x_1_3 1 9)
.....
(= (+ x_1_2 x_1_3) 5)
(alldifferent x_1_2 x_1_3)
.....
```

カックロを Sugar で解く

▸ Web

カックロの制約記述

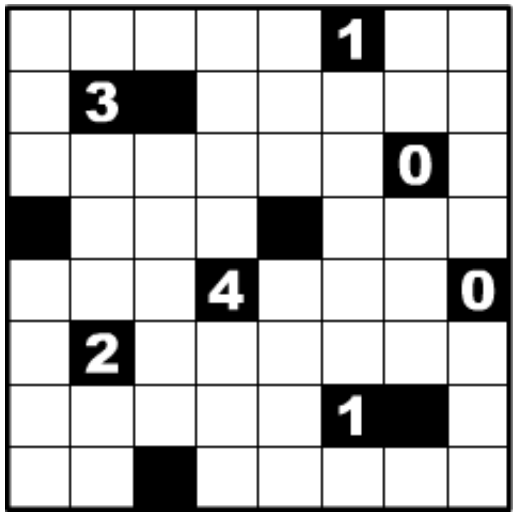
- *i* 行 *j* 列目の白マスに整数変数 x_*i*_*j* を割り当てる.
- 数字の合計の条件を記述する.
- 数字が相異なる条件を alldifferent で表す.

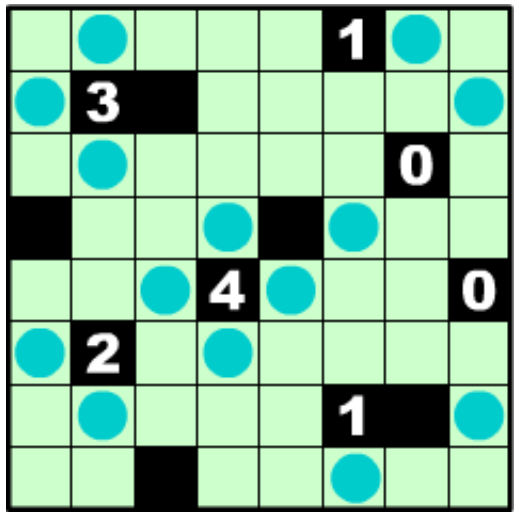
カックロの記述例

```
(int x_1_2 1 9)
(int x_1_3 1 9)
.....
(= (+ x_1_2 x_1_3) 5)
(alldifferent x_1_2 x_1_3)
.....
```

美術館

ニコリによる例題とルールの説明 ▸ Web





美術館のルール

1. 以下のルールに従って盤面の白マスに○ (照明) を配置します．
2. 数字は，タテヨコ両隣の最大 4 つの白マスに入る○の数を表します．
3. 照明は，そのマスから上下左右に，黒マスか外枠にぶつかるまでの範囲を照らします．ナナメには照らせません．
4. どの照明にも照らされてない白マスがあってはいけません．また，照明のあるマスは，他の照明で照らされてはいけません．

美術館を Sugar で解く

▸ Web

美術館の制約記述

- i 行 j 列目の白マスに整数変数 $\mathrm{x\_}i\mathrm{\_}j \in \{0, 1\}$ を割り当てる．1 が照明あり．
- 数字の回りの照明の数の条件を記述する．
- 上下あるいは左右に連続する白マス中にある照明の数が 1 以下，という条件を記述する．
- 各白マスについて，そこから上下および左右に連続する白マス中にある照明の数が 1 以上，という条件を記述する．

美術館の記述例

```
(int x_0_0 0 1)
.....
(= (+ x_0_4 x_0_6 x_1_5) 1)
.....
(<= (+ x_0_0 x_1_0 x_2_0) 1)
.....
(>= (+ x_2_3 x_1_3 x_0_3 x_2_2 x_2_1 x_2_0 x_2_4 x_2_5 x_3_3) 1)
.....
```

美術館を Sugar で解く

▸ Web

美術館の制約記述

- i 行 j 列目の白マスに整数変数 $x\_i\_j \in \{0, 1\}$ を割り当てる．1 が照明あり．
- 数字の回りの照明の数の条件を記述する．
- 上下あるいは左右に連続する白マス中にある照明の数が 1 以下，という条件を記述する．
- 各白マスについて，そこから上下および左右に連続する白マス中にある照明の数が 1 以上，という条件を記述する．

美術館の記述例

```
(int x_0_0 0 1)
.....
(= (+ x_0_4 x_0_6 x_1_5) 1)
.....
(<= (+ x_0_0 x_1_0 x_2_0) 1)
.....
(>= (+ x_2_3 x_1_3 x_0_3 x_2_2 x_2_1 x_2_0 x_2_4 x_2_5 x_3_3) 1)
.....
```

美術館を Sugar で解く

▸ Web

美術館の制約記述

- i 行 j 列目の白マスに整数変数 $\mathrm{x\_}i\mathrm{\_}j \in \{0, 1\}$ を割り当てる．1 が照明あり．
- 数字の回りの照明の数の条件を記述する．
- 上下あるいは左右に連続する白マス中にある照明の数が 1 以下，という条件を記述する．
- 各白マスについて，そこから上下および左右に連続する白マス中にある照明の数が 1 以上，という条件を記述する．

美術館の記述例

```
(int x_0_0 0 1)
.....
(= (+ x_0_4 x_0_6 x_1_5) 1)
.....
(<= (+ x_0_0 x_1_0 x_2_0) 1)
.....
(>= (+ x_2_3 x_1_3 x_0_3 x_2_2 x_2_1 x_2_0 x_2_4 x_2_5 x_3_3) 1)
.....
```

美術館を Sugar で解く

▸ Web

美術館の制約記述

- i 行 j 列目の白マスに整数変数 $\mathrm{x\_}i\mathrm{\_}j \in \{0, 1\}$ を割り当てる．1 が照明あり．
- 数字の回りの照明の数の条件を記述する．
- 上下あるいは左右に連続する白マス中にある照明の数が 1 以下，という条件を記述する．
- 各白マスについて，そこから上下および左右に連続する白マス中にある照明の数が 1 以上，という条件を記述する．

美術館の記述例

```
(int x_0_0 0 1)
.....
(= (+ x_0_4 x_0_6 x_1_5) 1)
.....
(<= (+ x_0_0 x_1_0 x_2_0) 1)
.....
(>= (+ x_2_3 x_1_3 x_0_3 x_2_2 x_2_1 x_2_0 x_2_4 x_2_5 x_3_3) 1)
.....
```

美術館を Sugar で解く

▸ Web

美術館の制約記述

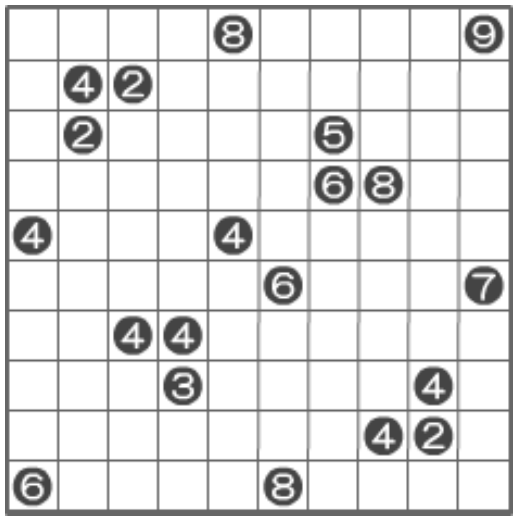
- i 行 j 列目の白マスに整数変数 $\mathrm{x\_}i\mathrm{\_}j \in \{0, 1\}$ を割り当てる．1 が照明あり．
- 数字の回りの照明の数の条件を記述する．
- 上下あるいは左右に連続する白マス中にある照明の数が 1 以下，という条件を記述する．
- 各白マスについて，そこから上下および左右に連続する白マス中にある照明の数が 1 以上，という条件を記述する．

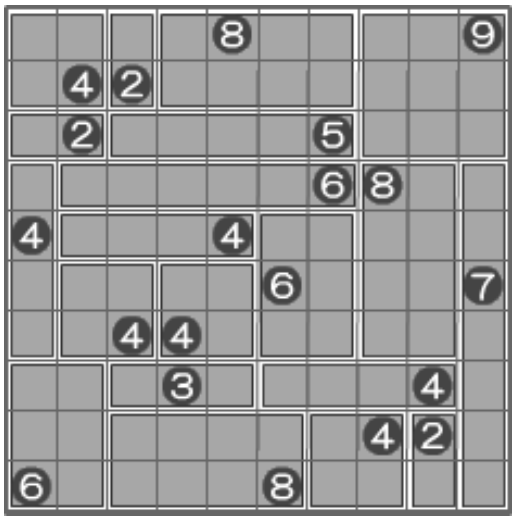
美術館の記述例

```
(int x_0_0 0 1)
.....
(= (+ x_0_4 x_0_6 x_1_5) 1)
.....
(<= (+ x_0_0 x_1_0 x_2_0) 1)
.....
(>= (+ x_2_3 x_1_3 x_0_3 x_2_2 x_2_1 x_2_0 x_2_4 x_2_5 x_3_3) 1)
.....
```

四角に切れ

ニコリによる例題とルールの説明 ▸ Web





四角に切れのルール

1. 盤面を長方形 (正方形) に分割します．
2. 数字は，1 マスの面積を 1 とした，長方形の面積です．4 と書いてあるマスを含む長方形は，1x4，2x2，4x1 のどれかになります．
3. 切るのは点線の上で，どの長方形にも数字が 1 つずつ入ります．

四角に切れを Sugar で解く

▸ Web

四角に切れの制約記述

- k 番目の長方形の位置を r_k, c_k で表し，高さと幅を h_k, w_k で表す．
- r_k, c_k, h_k, w_k の条件を記述する．
- 各数字が対応する長方形の内部にある条件を記述する．
- 長方形が互いに重ならない条件を記述する．

四角に切れの記述例

```
(int r_0 0 9) (int c_0 0 9)
(int h_0 (1 2 4 8))  (int w_0 (1 2 4 8))
(or (and (= h_0 1) (= w_0 8)) (and (= h_0 2) (= w_0 4))
    (and (= h_0 4) (= w_0 2)) (and (= h_0 8) (= w_0 1)))
(<= (+ r_0 h_0) 10) (<= (+ c_0 w_0) 10)
.....
(<= r_0 0) (< 0 (+ r_0 h_0)) (<= c_0 4) (< 4 (+ c_0 w_0))
.....
(or (<= (+ r_0 h_0) r_1) (<= (+ r_1 h_1) r_0)
    (<= (+ c_0 w_0) c_1) (<= (+ c_1 w_1) c_0))
.....
```

四角に切れを Sugar で解く

▸ Web

四角に切れの制約記述

- k 番目の長方形の位置を r_k, c_k で表し，高さと幅を h_k, w_k で表す．
- r_k, c_k, h_k, w_k の条件を記述する．
- 各数字が対応する長方形の内部にある条件を記述する．
- 長方形が互いに重ならない条件を記述する．

四角に切れの記述例

```
(int r_0 0 9) (int c_0 0 9)
(int h_0 (1 2 4 8))  (int w_0 (1 2 4 8))
(or (and (= h_0 1) (= w_0 8)) (and (= h_0 2) (= w_0 4))
    (and (= h_0 4) (= w_0 2)) (and (= h_0 8) (= w_0 1)))
(<= (+ r_0 h_0) 10) (<= (+ c_0 w_0) 10)
.....
(<= r_0 0) (< 0 (+ r_0 h_0)) (<= c_0 4) (< 4 (+ c_0 w_0))
.....
(or (<= (+ r_0 h_0) r_1) (<= (+ r_1 h_1) r_0)
    (<= (+ c_0 w_0) c_1) (<= (+ c_1 w_1) c_0))
.....
```

四角に切れを Sugar で解く

▸ Web

四角に切れの制約記述

- k 番目の長方形の位置を r_k, c_k で表し，高さと幅を h_k, w_k で表す．
- r_k, c_k, h_k, w_k の条件を記述する．
- 各数字が対応する長方形の内部にある条件を記述する．
- 長方形が互いに重ならない条件を記述する．

四角に切れの記述例

```
(int r_0 0 9) (int c_0 0 9)
(int h_0 (1 2 4 8))  (int w_0 (1 2 4 8))
(or (and (= h_0 1) (= w_0 8)) (and (= h_0 2) (= w_0 4))
    (and (= h_0 4) (= w_0 2)) (and (= h_0 8) (= w_0 1)))
(<= (+ r_0 h_0) 10) (<= (+ c_0 w_0) 10)
.....
(<= r_0 0) (< 0 (+ r_0 h_0)) (<= c_0 4) (< 4 (+ c_0 w_0))
.....
(or (<= (+ r_0 h_0) r_1) (<= (+ r_1 h_1) r_0)
    (<= (+ c_0 w_0) c_1) (<= (+ c_1 w_1) c_0))
.....
```

四角に切れを Sugar で解く

▸ Web

四角に切れの制約記述

- k 番目の長方形の位置を r_k, c_k で表し，高さと幅を h_k, w_k で表す．
- r_k, c_k, h_k, w_k の条件を記述する．
- 各数字が対応する長方形の内部にある条件を記述する．
- 長方形が互いに重ならない条件を記述する．

四角に切れの記述例

```
(int r_0 0 9) (int c_0 0 9)
(int h_0 (1 2 4 8))  (int w_0 (1 2 4 8))
(or (and (= h_0 1) (= w_0 8)) (and (= h_0 2) (= w_0 4))
    (and (= h_0 4) (= w_0 2)) (and (= h_0 8) (= w_0 1)))
(<= (+ r_0 h_0) 10) (<= (+ c_0 w_0) 10)
.....
(<= r_0 0) (< 0 (+ r_0 h_0)) (<= c_0 4) (< 4 (+ c_0 w_0))
.....
(or (<= (+ r_0 h_0) r_1) (<= (+ r_1 h_1) r_0)
    (<= (+ c_0 w_0) c_1) (<= (+ c_1 w_1) c_0))
.....
```

四角に切れを Sugar で解く

▸ Web

四角に切れの制約記述

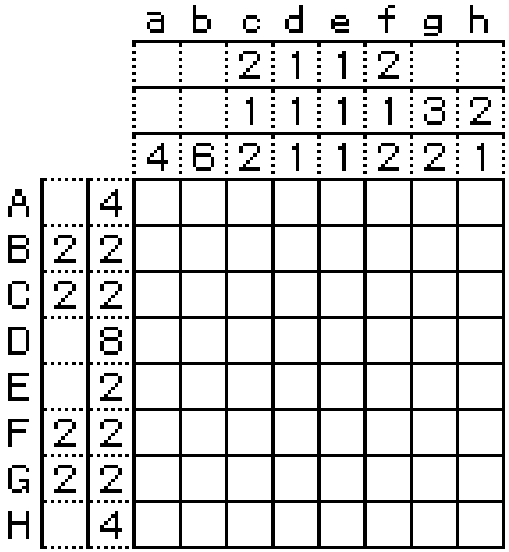
- k 番目の長方形の位置を r_k, c_k で表し，高さと幅を h_k, w_k で表す．
- r_k, c_k, h_k, w_k の条件を記述する．
- 各数字が対応する長方形の内部にある条件を記述する．
- 長方形が互いに重ならない条件を記述する．

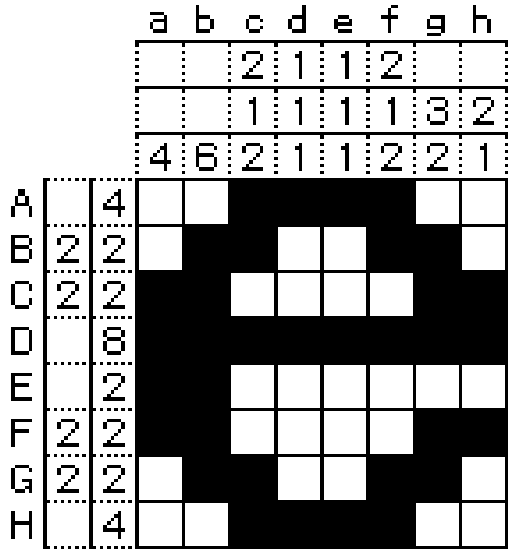
四角に切れの記述例

```
(int r_0 0 9) (int c_0 0 9)
(int h_0 (1 2 4 8))  (int w_0 (1 2 4 8))
(or (and (= h_0 1) (= w_0 8)) (and (= h_0 2) (= w_0 4))
    (and (= h_0 4) (= w_0 2)) (and (= h_0 8) (= w_0 1)))
(<= (+ r_0 h_0) 10) (<= (+ c_0 w_0) 10)
.....
(<= r_0 0) (< 0 (+ r_0 h_0)) (<= c_0 4) (< 4 (+ c_0 w_0))
.....
(or (<= (+ r_0 h_0) r_1) (<= (+ r_1 h_1) r_0)
    (<= (+ c_0 w_0) c_1) (<= (+ c_1 w_1) c_0))
.....
```

お絵かきロジック

Wikipedia による例題





お絵かきロジックのルール

1. 以下のルールに従って盤面をぬりつぶします．
2. 数字は，その行または列で連続に塗りつぶす黒マスのブロックの長さを表します．
3. 数字が複数ある場合，黒マスのブロックはその順序に並びます．また，黒マスのブロックどうしの間は 1 マス以上空いていなければなりません．

お絵かきロジックを Sugar で解く

▸ Web

お絵かきロジックの制約記述

- i 行 j 列目のマスに整数変数 $\mathrm{x}\_i\_j \in \{0, 1\}$ を割り当てる．
- i 行目 k 番目のブロックの位置を h_i_k で，j 列目 k 番目のブロックの位置を v_j_k で表す．
- i 行 j 列目のマスがぬりつぶされる条件を記述する．
- ブロックの順序の条件を記述する．

お絵かきロジックの記述例

```
(int x_0_0 0 1)
.....
(int h_0_0 0 4)
(iff (= x_0_0 1) (and (<= h_0_0 0) (< 0 (+ h_0_0 4))))
.....
(< (+ h_1_0 2) h_1_1)
.....
```

お絵かきロジックを Sugar で解く

Web

お絵かきロジックの制約記述

- *i* 行 *j* 列目のマスに整数変数 $x\_i\_j \in \{0,1\}$ を割り当てる．
- *i* 行目 *k* 番目のブロックの位置を h_i_k で，*j* 列目 *k* 番目のブロックの位置を v_j_k で表す．
- *i* 行 *j* 列目のマスがぬりつぶされる条件を記述する．
- ブロックの順序の条件を記述する．

お絵かきロジックの記述例

```
(int x_0_0 0 1)
.....
(int h_0_0 0 4)
(iff (= x_0_0 1) (and (<= h_0_0 0) (< 0 (+ h_0_0 4))))
.....
(< (+ h_1_0 2) h_1_1)
.....
```

お絵かきロジックを Sugar で解く

Web

お絵かきロジックの制約記述

- i 行 j 列目のマスに整数変数 $\mathrm{x\_}i\mathrm{\_}j \in \{0,1\}$ を割り当てる．
- i 行目 k 番目のブロックの位置を $\mathrm{h\_}i\mathrm{\_}k$ で，j 列目 k 番目のブロックの位置を $\mathrm{v\_}j\mathrm{\_}k$ で表す．
- i 行 j 列目のマスがぬりつぶされる条件を記述する．
- ブロックの順序の条件を記述する．

お絵かきロジックの記述例

```
(int x_0_0 0 1)
.....
(int h_0_0 0 4)
(iff (= x_0_0 1) (and (<= h_0_0 0) (< 0 (+ h_0_0 4))))
.....
(< (+ h_1_0 2) h_1_1)
.....
```

お絵かきロジックを Sugar で解く

▸ Web

お絵かきロジックの制約記述

- i 行 j 列目のマスに整数変数 $\mathrm{x\_}i\mathrm{\_}j \in \{0,1\}$ を割り当てる．
- i 行目 k 番目のブロックの位置を $\mathrm{h\_}i\mathrm{\_}k$ で，j 列目 k 番目のブロックの位置を $\mathrm{v\_}j\mathrm{\_}k$ で表す．
- i 行 j 列目のマスがぬりつぶされる条件を記述する．
- ブロックの順序の条件を記述する．

お絵かきロジックの記述例

```
(int x_0_0 0 1)
.....
(int h_0_0 0 4)
(iff (= x_0_0 1) (and (<= h_0_0 0) (< 0 (+ h_0_0 4))))
.....
(< (+ h_1_0 2) h_1_1)
.....
```

お絵かきロジックを Sugar で解く

▸ Web

お絵かきロジックの制約記述

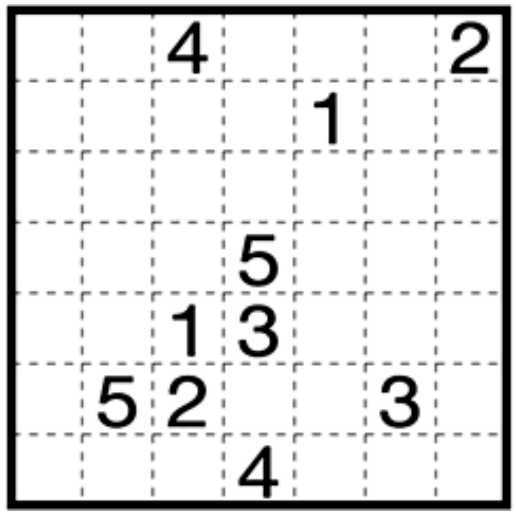
- i 行 j 列目のマスに整数変数 $\mathrm{x\_}i\mathrm{\_}j \in \{0, 1\}$ を割り当てる．
- i 行目 k 番目のブロックの位置を $\mathrm{h\_}i\mathrm{\_}k$ で，j 列目 k 番目のブロックの位置を $\mathrm{v\_}j\mathrm{\_}k$ で表す．
- i 行 j 列目のマスがぬりつぶされる条件を記述する．
- ブロックの順序の条件を記述する．

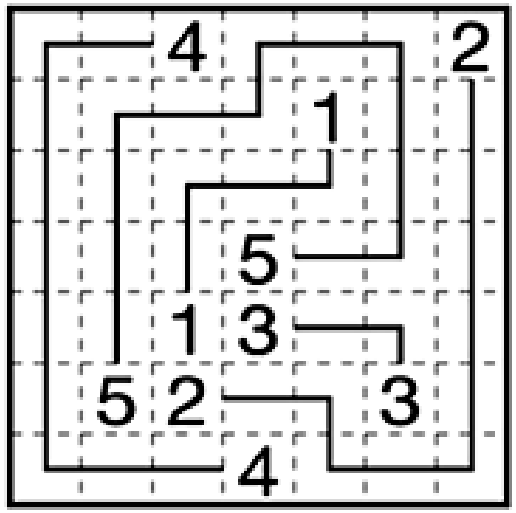
お絵かきロジックの記述例

```
(int x_0_0 0 1)
.....
(int h_0_0 0 4)
(iff (= x_0_0 1) (and (<= h_0_0 0) (< 0 (+ h_0_0 4))))
.....
(< (+ h_1_0 2) h_1_1)
.....
```

ナンバーリンク

ニコリによる例題とルールの説明 ▸ Web





ナンバーリンクのルール

1. 同じ数字どうしを線でつなげます．
2. 線はタテヨコに引き，マスの中央を通ります．
3. 線は 1 マスに 1 本だけ通過できます．線をワクの外に出したり，交差や枝分かれさせてはいけません．また，線は数字が入っているマスを通過してもいけません．

ナンバーリンクを Sugar で解く

▸ Web

ナンバーリンクの制約記述

- i 行 j 列目のマスから下への辺を $\mathrm{v}\_i\_j \in \{0,1\}$，右への辺を $\mathrm{h}\_i\_j \in \{0,1\}$ で表す．
- 各マスについて，次数 (degree) の条件を記述する．数字マスの次数は 1，白マスの次数は 0 または 2 である．
- i 行 j 列目のマスがどの数字とつながっているかを変数 x_i_j で表す．辺がつながっていれば x_i_j の値が等しくなる条件を記述する．

ナンバーリンクの記述例

```
(int v_0_0 0 1)  (int h_0_0 0 1)
.....
(int d_0_0 (0 2))  (= d_0_0 (+ v_0_0 h_0_0))
.....
(int x_0_0 1 5)
(=> (> v_0_0 0) (= x_0_0 x_1_0))
.....
```

ナンバーリンクを Sugar で解く

▸ Web

ナンバーリンクの制約記述

- **i 行 j 列目のマスから下への辺を $\mathrm{v}\_i\_j \in \{0,1\}$，右への辺を $\mathrm{h}\_i\_j \in \{0,1\}$ で表す．**
- **各マスについて，次数 (degree) の条件を記述する．数字マスの次数は 1，白マスの次数は 0 または 2 である．**
- **i 行 j 列目のマスがどの数字とつながっているかを変数 x_i_j で表す．辺がつながっていれば x_i_j の値が等しくなる条件を記述する．**

ナンバーリンクの記述例

```
(int v_0_0 0 1)  (int h_0_0 0 1)
.....
(int d_0_0 (0 2))  (= d_0_0 (+ v_0_0 h_0_0))
.....
(int x_0_0 1 5)
(=> (> v_0_0 0) (= x_0_0 x_1_0))
.....
```

ナンバーリンクを Sugar で解く

▸ Web

ナンバーリンクの制約記述

- i 行 j 列目のマスから下への辺を $\mathrm{v}\_i\_j \in \{0,1\}$，右への辺を $\mathrm{h}\_i\_j \in \{0,1\}$ で表す．
- 各マスについて，次数 (degree) の条件を記述する．数字マスの次数は 1，白マスの次数は 0 または 2 である．
- i 行 j 列目のマスがどの数字とつながっているかを変数 x_i_j で表す．辺がつながっていれば x_i_j の値が等しくなる条件を記述する．

ナンバーリンクの記述例

```
(int v_0_0 0 1)  (int h_0_0 0 1)
.....
(int d_0_0 (0 2))  (= d_0_0 (+ v_0_0 h_0_0))
.....
(int x_0_0 1 5)
(=> (> v_0_0 0) (= x_0_0 x_1_0))
.....
```

ナンバーリンクを Sugar で解く

▸ Web

ナンバーリンクの制約記述

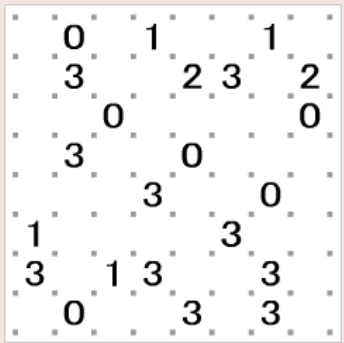
- *i* 行 *j* 列目のマスから下への辺を $\mathrm{v\_}i\mathrm{\_}j \in \{0,1\}$，右への辺を $\mathrm{h\_}i\mathrm{\_}j \in \{0,1\}$ で表す．
- 各マスについて，次数 (degree) の条件を記述する．数字マスの次数は 1，白マスの次数は 0 または 2 である．
- *i* 行 *j* 列目のマスがどの数字とつながっているかを変数 $\mathrm{x\_}i\mathrm{\_}j$ で表す．辺がつながっていれば $\mathrm{x\_}i\mathrm{\_}j$ の値が等しくなる条件を記述する．

ナンバーリンクの記述例

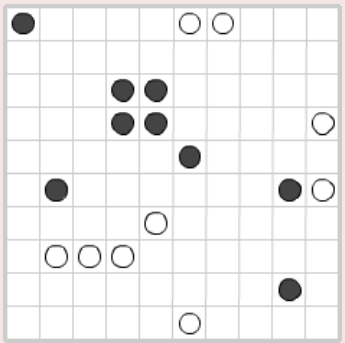
```
(int v_0_0 0 1)  (int h_0_0 0 1)
.....
(int d_0_0 (0 2))  (= d_0_0 (+ v_0_0 h_0_0))
.....
(int x_0_0 1 5)
(=> (> v_0_0 0) (= x_0_0 x_1_0))
.....
```

単一ループの条件が必要なパズル

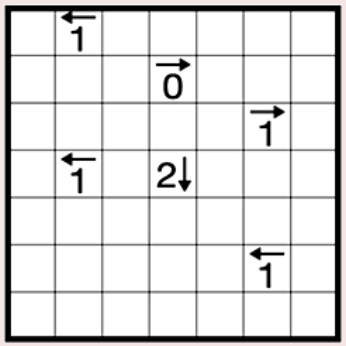
スリザーリンク



ましゅ

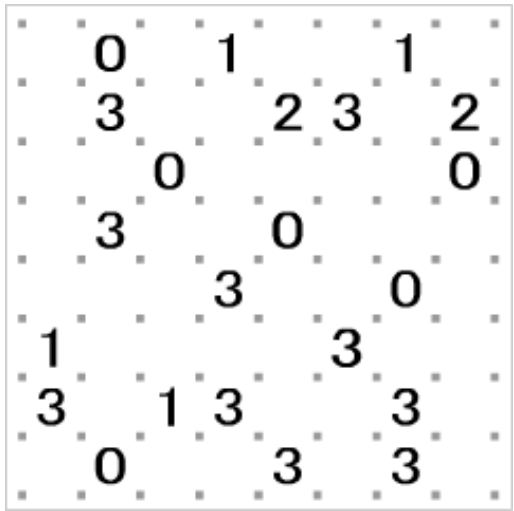


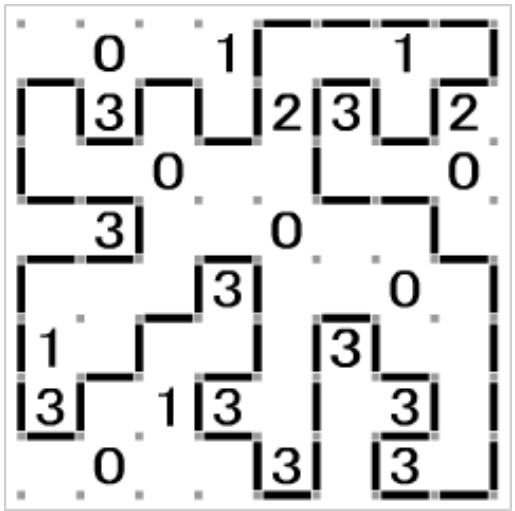
ヤジリン



スリザーリンク

ニコリによる例題とルールの説明 ▸ Web





スリザーリンクのルール

1. 点と点をタテヨコにつなげ，全体で 1 つの輪っかを作ります．
2. 4 つの点で作られた小さな正方形の中の数字は，その正方形の辺に引く線の数です．数字のない小さな正方形の辺には，何本の線を引くかわかりません．
3. 線は，交差したり枝分かれしたりはしません．

スリザーリンクを Sugar で解く (1)

Web

「全体で 1 つの輪っか」という単一ループの条件の記述が難しい．

スリザーリンクの制約記述 (単一ループの条件以外)

- i 行 j 列目のマスの左の辺を v_i_j，右の辺を h_i_j で表す．後述の条件記述のため，辺の向きを考慮してドメインは $\{-1, 0, 1\}$ とする．
- 各数字マスの回りの辺の個数の条件を記述する．
- 各節点における次数が 0 または 2 となる条件，および入次数と出次数が一致する条件を記述する．

スリザーリンクの記述例 (1)

```
(int v_0_0 -1 1) (int h_0_0 -1 1)
.....
(= (+ (abs v_3_1) (abs h_3_1) (abs v_3_2) (abs h_4_1)) 3)
.....
(int d_1_2 (0 2))
(= d_1_2 (+ (abs v_0_2) (abs h_1_1) (abs v_1_2) (abs h_1_2)))
(= (+ v_0_2 h_1_1 (neg v_1_2) (neg h_1_2)) 0)
.....
```

スリザーリンクを Sugar で解く (1)

Web

「全体で 1 つの輪っか」という単一ループの条件の記述が難しい．

スリザーリンクの制約記述 (単一ループの条件以外)

- *i* 行 *j* 列目のマスの左の辺を v_*i*_*j*，右の辺を h_*i*_*j* で表す．後述の条件記述のため，辺の向きを考慮してドメインは $\{-1, 0, 1\}$ とする．
- 各数字マスの回りの辺の個数の条件を記述する．
- 各節点における次数が 0 または 2 となる条件，および入次数と出次数が一致する条件を記述する．

スリザーリンクの記述例 (1)

```
(int v_0_0 -1 1) (int h_0_0 -1 1)
.....
(= (+ (abs v_3_1) (abs h_3_1) (abs v_3_2) (abs h_4_1)) 3)
.....
(int d_1_2 (0 2))
(= d_1_2 (+ (abs v_0_2) (abs h_1_1) (abs v_1_2) (abs h_1_2)))
(= (+ v_0_2 h_1_1 (neg v_1_2) (neg h_1_2)) 0)
.....
```

スリザーリンクを Sugar で解く (1)

Web

「全体で 1 つの輪っか」という単一ループの条件の記述が難しい.

スリザーリンクの制約記述 (単一ループの条件以外)

- *i* 行 *j* 列目のマスの左の辺を v_*i*_*j*，右の辺を h_*i*_*j* で表す．後述の条件記述のため，辺の向きを考慮してドメインは $\{-1, 0, 1\}$ とする.
- 各数字マスの回りの辺の個数の条件を記述する.
- 各節点における次数が 0 または 2 となる条件，および入次数と出次数が一致する条件を記述する.

スリザーリンクの記述例 (1)

```
(int v_0_0 -1 1) (int h_0_0 -1 1)
.....
(= (+ (abs v_3_1) (abs h_3_1) (abs v_3_2) (abs h_4_1)) 3)
.....
(int d_1_2 (0 2))
(= d_1_2 (+ (abs v_0_2) (abs h_1_1) (abs v_1_2) (abs h_1_2)))
(= (+ v_0_2 h_1_1 (neg v_1_2) (neg h_1_2)) 0)
.....
```

スリザーリンクを Sugar で解く (1)

Web

「全体で 1 つの輪っか」という単一ループの条件の記述が難しい.

スリザーリンクの制約記述 (単一ループの条件以外)

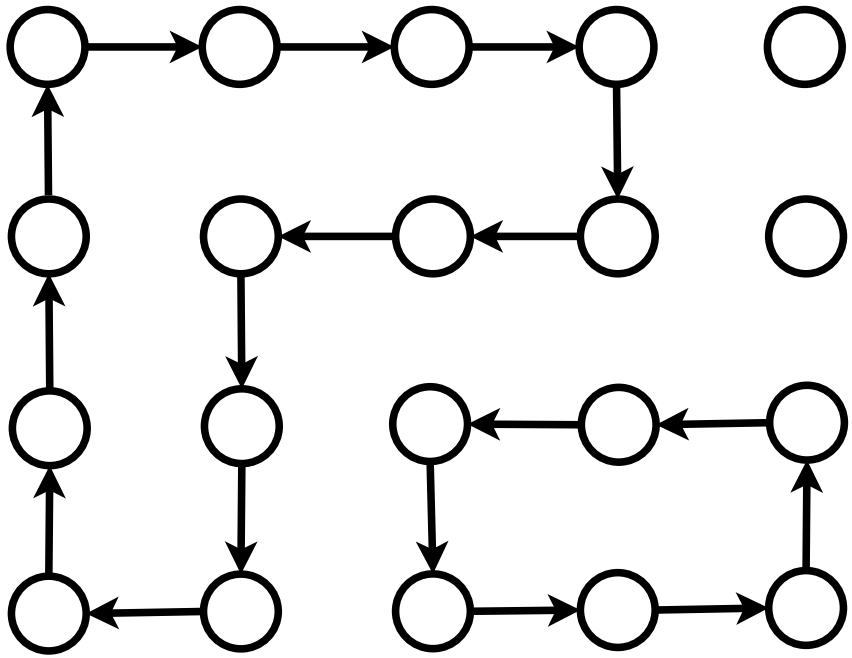
- *i* 行 *j* 列目のマスの左の辺を v_*i*_*j*，右の辺を h_*i*_*j* で表す．後述の条件記述のため，辺の向きを考慮してドメインは $\{-1, 0, 1\}$ とする.
- 各数字マスの回りの辺の個数の条件を記述する.
- 各節点における次数が 0 または 2 となる条件，および入次数と出次数が一致する条件を記述する.

スリザーリンクの記述例 (1)

```
(int v_0_0 -1 1) (int h_0_0 -1 1)
.....
(= (+ (abs v_3_1) (abs h_3_1) (abs v_3_2) (abs h_4_1)) 3)
.....
(int d_1_2 (0 2))
(= d_1_2 (+ (abs v_0_2) (abs h_1_1) (abs v_1_2) (abs h_1_2)))
(= (+ v_0_2 h_1_1 (neg v_1_2) (neg h_1_2)) 0)
.....
```

スリザーリンクを Sugar で解く (2)

▸ Web

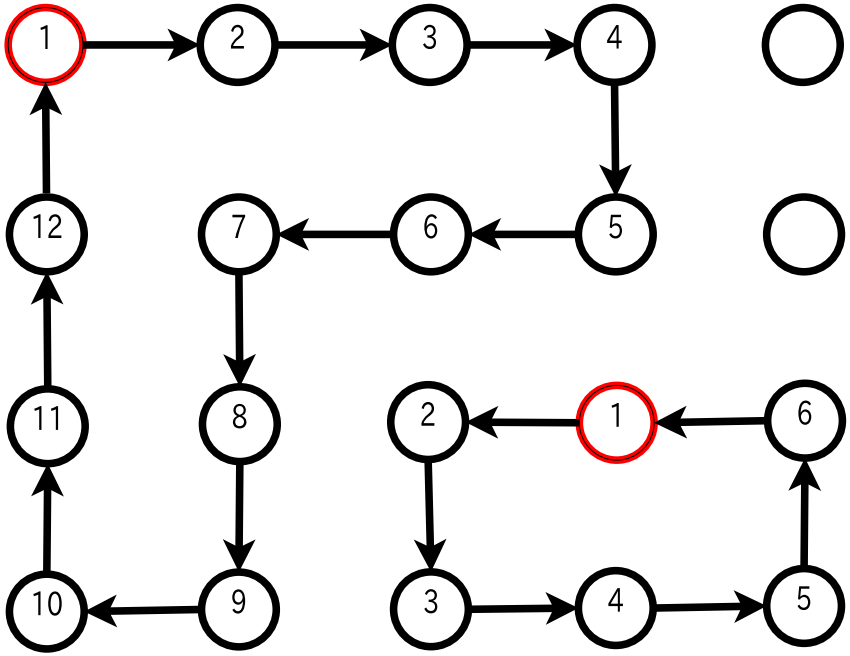


「全体で 1 つの輪っかを作る」には

- 上のように複数のループができる可能性がある.

スリザーリンクを Sugar で解く (2)

Web



「全体で 1 つの輪っかを作る」には

- 上のように複数のループができる可能性がある.
- そこで，辺の向きに沿って節点に順に番号を付ける.
- 番号 1 の付く節点 (始節点) の個数が，盤面全体で 1 つであれば良い.

スリザーリンクを Sugar で解く (3)

Web

単一ループの制約記述

- 節点の番号を表す変数を x_i_j，始節点かどうかを表す変数を y_i_j ∈ {0, 1} とする．
- 辺の向きに番号が増える条件を記述する．
- 盤面全体で始節点が 1 つ，という条件を記述する．

スリザーリンクの記述例 (2)

```
(int x_0_0 0 81)
(int y_0_0 0 1)
(iff (> d_0_0 0) (> x_0_0 0))
(iff (= x_0_0 1) (= y_0_0 1))
.....
(=> (> v_0_0 0) (or (> x_1_0 x_0_0) (= x_1_0 1)))
(=> (> h_0_0 0) (or (> x_0_1 x_0_0) (= x_0_1 1)))
.....
(= (+ y_0_0 y_0_1 ... y_8_8) 1)
```

スリザーリンクを Sugar で解く (3)

Web

単一ループの制約記述

- **節点の番号を表す変数を x_i_j，始節点かどうかを表す変数を $y\_i\_j \in \{0,1\}$ とする．**
- **辺の向きに番号が増える条件を記述する．**
- **盤面全体で始節点が 1 つ，という条件を記述する．**

スリザーリンクの記述例 (2)

```
(int x_0_0 0 81)
(int y_0_0 0 1)
(iff (> d_0_0 0) (> x_0_0 0))
(iff (= x_0_0 1) (= y_0_0 1))
.....
(=> (> v_0_0 0) (or (> x_1_0 x_0_0) (= x_1_0 1)))
(=> (> h_0_0 0) (or (> x_0_1 x_0_0) (= x_0_1 1)))
.....
(= (+ y_0_0 y_0_1 ... y_8_8) 1)
```

スリザーリンクを Sugar で解く (3)

Web

単一ループの制約記述

- 節点の番号を表す変数を x_i_j，始節点かどうかを表す変数を $y\_i\_j \in \{0,1\}$ とする．
- 辺の向きに番号が増える条件を記述する．
- 盤面全体で始節点が 1 つ，という条件を記述する．

スリザーリンクの記述例 (2)

```
(int x_0_0 0 81)
(int y_0_0 0 1)
(iff (> d_0_0 0) (> x_0_0 0))
(iff (= x_0_0 1) (= y_0_0 1))
.....
(=> (> v_0_0 0) (or (> x_1_0 x_0_0) (= x_1_0 1)))
(=> (> h_0_0 0) (or (> x_0_1 x_0_0) (= x_0_1 1)))
.....
(= (+ y_0_0 y_0_1 ... y_8_8) 1)
```

スリザーリンクを Sugar で解く (3)

Web

単一ループの制約記述

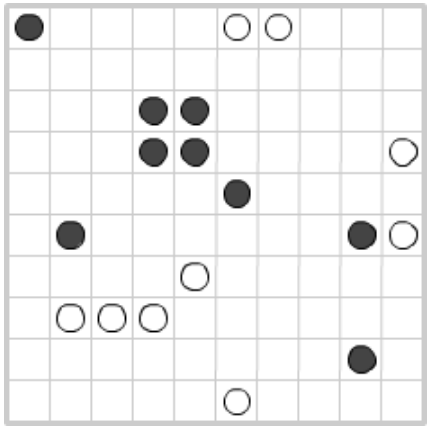
- 節点の番号を表す変数を x_i_j，始節点かどうかを表す変数を y_i_j ∈ {0, 1} とする．
- 辺の向きに番号が増える条件を記述する．
- 盤面全体で始節点が 1 つ，という条件を記述する．

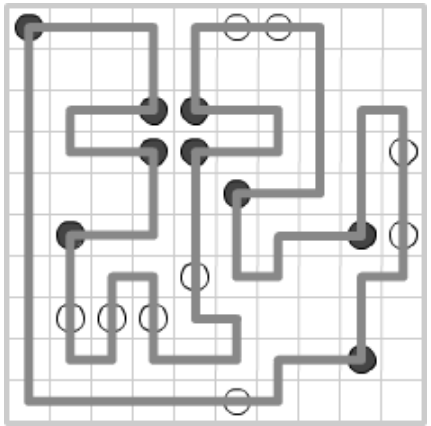
スリザーリンクの記述例 (2)

```
(int x_0_0 0 81)
(int y_0_0 0 1)
(iff (> d_0_0 0) (> x_0_0 0))
(iff (= x_0_0 1) (= y_0_0 1))
.....
(=> (> v_0_0 0) (or (> x_1_0 x_0_0) (= x_1_0 1)))
(=> (> h_0_0 0) (or (> x_0_1 x_0_0) (= x_0_1 1)))
.....
(= (+ y_0_0 y_0_1 ... y_8_8) 1)
```

ましゅ

ニコリによる例題とルールの説明 ▸ Web





ましゅのルール

1. 盤面に線を引き，全体で 1 つの輪っかを作ります．輪っかを作る線はタテヨコにマスの中央を通り，1 マスに 1 本だけ通過できます．線をワクの外に出したり，交差や枝分かれさせてはいけません．
2. 白丸・黒丸があるマスは必ず線が通ります．
3. 白丸を通る線は，白丸のマスで必ず直進し，白丸の両隣のマスの少なくとも片方で直角に曲がります．
4. 黒丸を通る線は，黒丸のマスで必ず直角に曲がりますが，黒丸の隣のマスで曲がってはいけません．

ましゅを Sugar で解く

Web

ましゅの制約記述

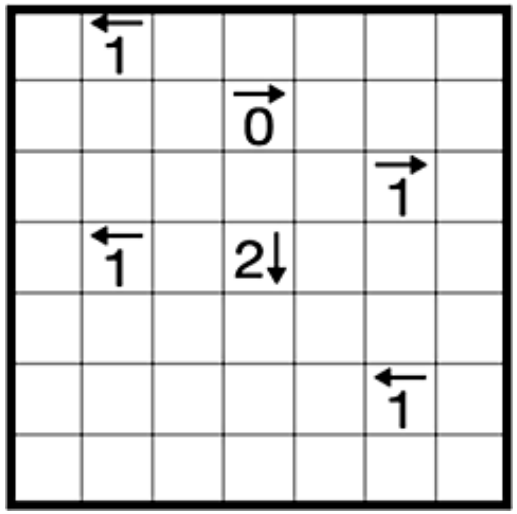
- *i* 行 *j* 列目のマスで，線が直進しているか曲がっているかを表すブール変数 s_*i*_*j* を用意し，白丸，黒丸についての条件を記述する．
- 単一ループ条件については，スリザーリンクと同様に考えれば良い．

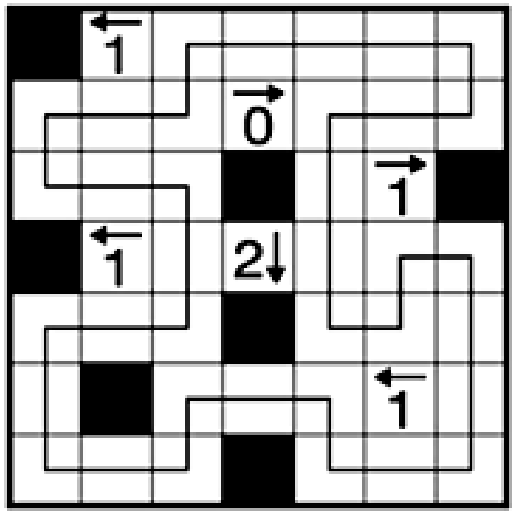
ましゅの記述例

```
(bool s_2_3)
(iff s_2_3 (or (and (!= v_1_3 0) (!= v_2_3 0))
               (and (!= h_2_2 0) (!= h_2_3 0))))
.....
s_6_4          ; 白丸
(=> (!= v_6_4 0) (or (not s_5_4) (not s_7_4)))
(=> (!= h_6_4 0) (or (not s_6_3) (not s_6_5)))
.....
(not s_2_3)   ; 黒丸
(=> (!= v_1_3 0) s_1_3) (=> (!= v_2_3 0) s_3_3)
(=> (!= h_2_2 0) s_2_2) (=> (!= h_2_3 0) s_2_4)
```

ヤジリン

ニコリによる例題とルールの説明 ▸ Web





ヤジリンのルール

1. 盤面に線を引き，全体で 1 つの輪っかを作ります．
2. 線はタテヨコにマスの中央を通り，1 マスに 1 本だけ通過できます．線は交差や枝分かれはしません．
3. 線の通らないマスは黒マスになります．黒マスはタテヨコに連続しません．数字のマスに線は通りませんが，黒マスにはなりません．
4. 数字は矢印の方向に入る黒マスの数です．

ヤジリンを Sugar で解く

▸ Web

ヤジリンの制約記述

- 黒マスを変数 $\mathrm{x\_}i\mathrm{\_}j \in \{0,1\}$ で表し，黒マスが連続しない条件を記述する．
- 各マスについて，次数が $\mathrm{x\_}i\mathrm{\_}j$ の 2 倍に一致する条件，および入次数と出次数が一致する条件を記述する．
- 黒マスの数の条件を記述する．
- 単一ループ条件については，スリザーリンクと同様に考えれば良い．

ヤジリンの記述例

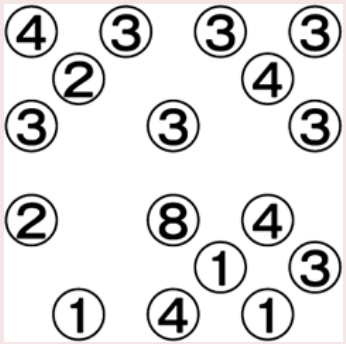
```
(int x_0_0 0 1)
(int x_1_0 0 1)
(or (= x_0_0 0) (= x_1_0 0))
.....
(= (+ (neg v_0_0) v_1_0 h_1_0) 0)
(= (+ (abs v_0_0) (abs v_1_0) (abs h_1_0)) (* 2 (- 1 x_1_0)))
.....
(= (+ x_4_3 x_5_3 x_6_3) 2)
.....
```

連結条件が必要なパズル

ひとりにしてくれ

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

橋をかけろ



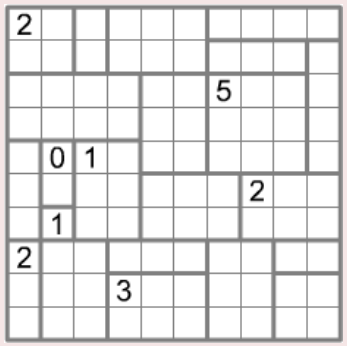
ぬりかべ

	3				1	
2			1			
	1			2		
		2				
1				1		6

フィルオミノ

	2		4		2	
1		2		6		6
3			3			3
			5			
3			2			3
3		2		4		2
	3		3		1	

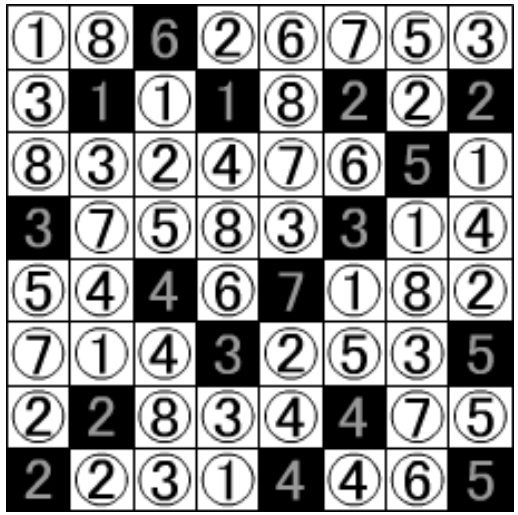
へやわけ



ひとりにしてくれ

ニコリによる例題とルールの説明 ▸ Web

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



ひとりにしてくれのルール

1. 盤面に並んでいる数字のうち，余計なものを黒くぬって，タテ列，ヨコ列に同じ数字が重複しないようにします．
2. 黒マスはタテヨコに連続しません．また，黒マスによって盤面が分断されることはありません．

ひとりにしてくれを Sugar で解く (1)

▸ Web

「黒マスによって盤面が分断されない」すなわち
「白マスが連結している」という条件の記述が難しい.

ひとりにしてくれの制約記述 (白マスの連結条件以外)

- i 行 j 列目を黒にするかどうかを $x\_i\_j \in \{0, 1\}$ で表す (0 が黒).
- 行および列に同じ数字が重複しない条件を記述する.
- 黒マスが連続しない条件を記述する.

ひとりにしてくれの記述例 (1)

```
(int x_0_0 0 1)
.....
(or (= x_0_2 0) (= x_0_4 0))
.....
(or (> x_0_0 0) (> x_1_0 0))
.....
```

ひとりにしてくれを Sugar で解く (1) ▸ Web

「黒マスによって盤面が分断されない」すなわち
「白マスが連結している」という条件の記述が難しい.

ひとりにしてくれの制約記述 (白マスの連結条件以外)

- i 行 j 列目を黒にするかどうかを $x\_i\_j \in \{0, 1\}$ で表す (0 が黒).
- 行および列に同じ数字が重複しない条件を記述する.
- 黒マスが連続しない条件を記述する.

ひとりにしてくれの記述例 (1)

```
(int x_0_0 0 1)
.....
(or (= x_0_2 0) (= x_0_4 0))
.....
(or (> x_0_0 0) (> x_1_0 0))
.....
```

ひとりにしてくれを Sugar で解く (1) ▸ Web

「黒マスによって盤面が分断されない」すなわち
「白マスが連結している」という条件の記述が難しい.

ひとりにしてくれの制約記述 (白マスの連結条件以外)

- i 行 j 列目を黒にするかどうかを $\mathrm{x\_}i\mathrm{\_}j \in \{0,1\}$ で表す (0 が黒).
- 行および列に同じ数字が重複しない条件を記述する.
- 黒マスが連続しない条件を記述する.

ひとりにしてくれの記述例 (1)

```
(int x_0_0 0 1)
.....
(or (= x_0_2 0) (= x_0_4 0))
.....
(or (> x_0_0 0) (> x_1_0 0))
.....
```

ひとりにしてくれを Sugar で解く (1) ▸ Web

「黒マスによって盤面が分断されない」すなわち
「白マスが連結している」という条件の記述が難しい.

ひとりにしてくれの制約記述 (白マスの連結条件以外)

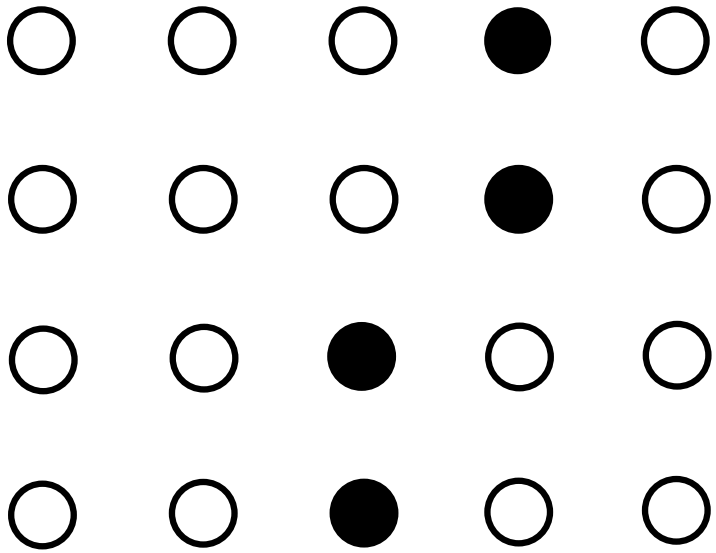
- i 行 j 列目を黒にするかどうかを $x\_i\_j \in \{0, 1\}$ で表す (0 が黒).
- 行および列に同じ数字が重複しない条件を記述する.
- 黒マスが連続しない条件を記述する.

ひとりにしてくれの記述例 (1)

```
(int x_0_0 0 1)
.....
(or (= x_0_2 0) (= x_0_4 0))
.....
(or (> x_0_0 0) (> x_1_0 0))
.....
```

ひとりにしてくれを Sugar で解く (2)

Web

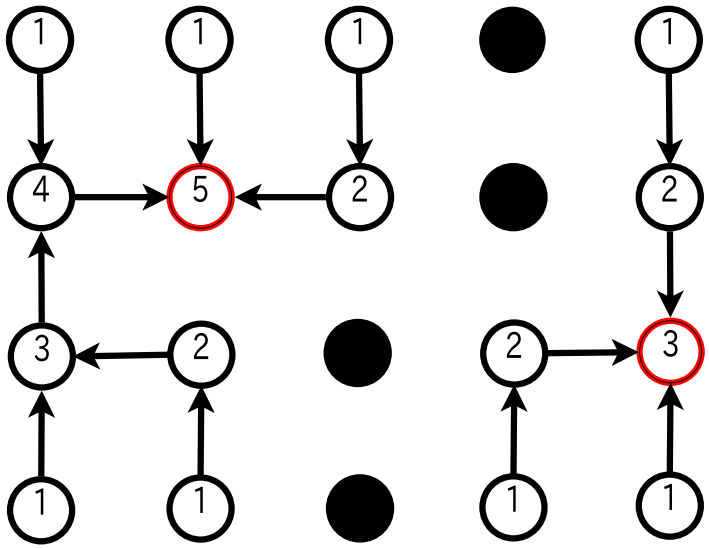


領域の連結条件の記述

- 上のように複数の領域ができる可能性がある.

ひとりにしてくれを Sugar で解く (2)

▸ Web



領域の連結条件の記述

- 上のように複数の領域ができる可能性がある．
- そこで，各領域のスパニング木 (spanning tree) を考える．
- 根に向かって増加するように節点に番号を付け，根の個数が盤面全体で 1 つになるようにできれば良い．

ひとりにしてくれを Sugar で解く (3) ▸ Web

白マスの連結条件の制約記述

- 白マスの番号を表す変数を z_i_j，根かどうかを表す変数を r_i_j $\in \{0,1\}$ とする．
- 根でない白マスならば番号が増加する頂点が存在する，という条件を記述する．
- 盤面全体で根が 1 つ，という条件を記述する．

ひとりにしてくれの記述例 (2)

```
(int z_0_0 0 64) (int r_0_0 0 1)
(iff (= x_0_0 0) (= z_0_0 0))
.....
(=> (and (> z_0_0 0) (<= r_0_0 0)) (or (< z_0_0 z_1_0) (< z_0_0 z_0_1)))
.....
(= (+ r_0_0 r_0_1 ... r_7_7) 1)
```

ひとりにしてくれを Sugar で解く (3)

Web

白マスの連結条件の制約記述

- **白マスの番号を表す変数を z_i_j，根かどうかを表す変数を $r\_i\_j \in \{0,1\}$ とする．**
- **根でない白マスならば番号が増加する頂点が存在する，という条件を記述する．**
- **盤面全体で根が 1 つ，という条件を記述する．**

ひとりにしてくれの記述例 (2)

```
(int z_0_0 0 64) (int r_0_0 0 1)
(iff (= x_0_0 0) (= z_0_0 0))
.....
(=> (and (> z_0_0 0) (<= r_0_0 0)) (or (< z_0_0 z_1_0) (< z_0_0 z_0_1)))
.....
(= (+ r_0_0 r_0_1 ... r_7_7) 1)
```

ひとりにしてくれを Sugar で解く (3)

Web

白マスの連結条件の制約記述

- 白マスの番号を表す変数を z_i_j，根かどうかを表す変数を r_i_j $\in \{0,1\}$ とする．
- 根でない白マスならば番号が増加する頂点が存在する，という条件を記述する．
- 盤面全体で根が 1 つ，という条件を記述する．

ひとりにしてくれの記述例 (2)

```
(int z_0_0 0 64) (int r_0_0 0 1)
(iff (= x_0_0 0) (= z_0_0 0))
.....
(=> (and (> z_0_0 0) (<= r_0_0 0)) (or (< z_0_0 z_1_0) (< z_0_0 z_0_1)))
.....
(= (+ r_0_0 r_0_1 ... r_7_7) 1)
```

ひとりにしてくれを Sugar で解く (3) ▸ Web

白マスの連結条件の制約記述

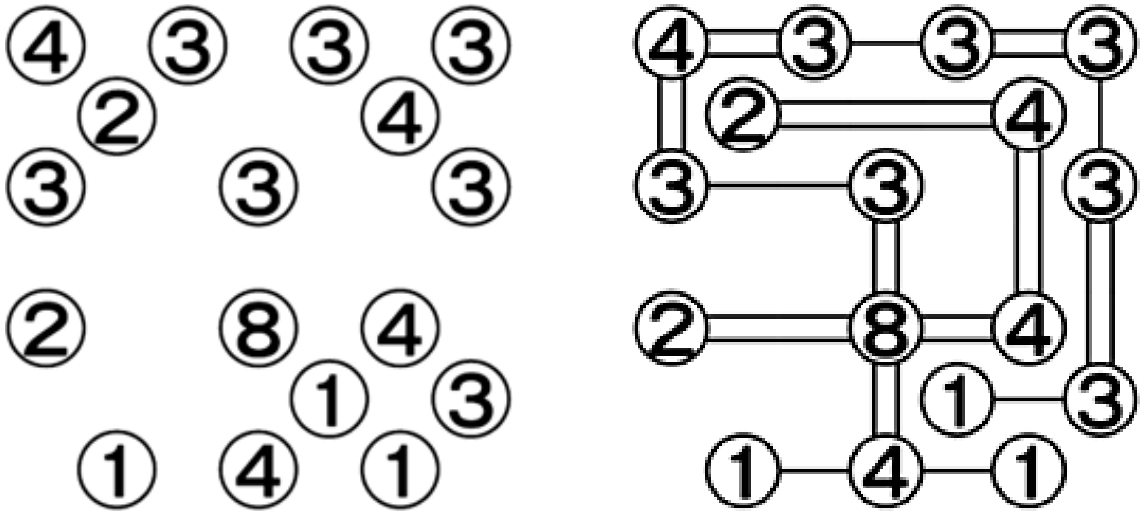
- 白マスの番号を表す変数を z_i_j，根かどうかを表す変数を r_i_j$\in \{0,1\}$ とする．
- 根でない白マスならば番号が増加する頂点が存在する，という条件を記述する．
- 盤面全体で根が 1 つ，という条件を記述する．

ひとりにしてくれの記述例 (2)

```
(int z_0_0 0 64) (int r_0_0 0 1)
(iff (= x_0_0 0) (= z_0_0 0))
.....
(=> (and (> z_0_0 0) (<= r_0_0 0)) (or (< z_0_0 z_1_0) (< z_0_0 z_0_1)))
.....
(= (+ r_0_0 r_0_1 ... r_7_7) 1)
```

橋をかけろ

ニコリによる例題とルールの説明 ▸ Web



橋をかけろのルール

1. 数字から数字に橋をかけて，全体をつなげます．
2. 数字は，その数字につながる橋の数です．
3. 橋はタテヨコにかけます．ナナメや，数字をとびこえてはかけられません．
4. 橋は 1 カ所に 2 本までかけられます．
5. 橋どうしが交差してはいけません．

橋をかけろを Sugar で解く

▸ Web

橋をかけろの制約記述

- 各橋を表す変数を v_*i*_*j*, h_*i*_*j* とする．ドメインは辺の向きと本数を含めて $\{-2, -1, 0, 1, 2\}$ とする．
- 橋の本数に関しての条件を記述する．
- 橋の交差に関しての条件を記述する．
- 橋の連結条件については，ひとりにしてくれと同様にする．

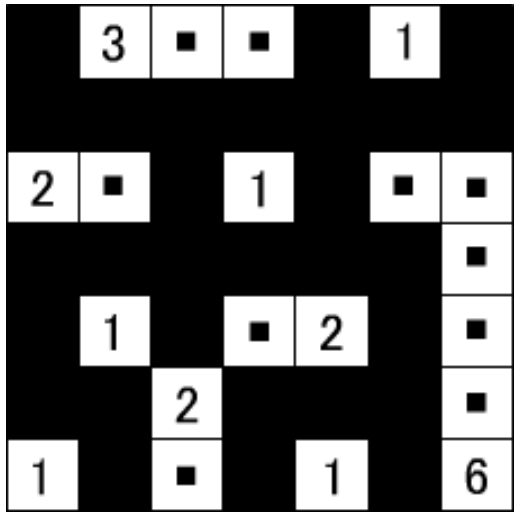
橋をかけろの記述例

```
(int v_0_0 -2 2)
(int h_0_0 -2 2)
.....
(= (+ (abs v_4_3) (abs h_4_3) (abs v_2_3) (abs h_4_0)) 8)
.....
(or (= v_1_1 0) (= h_2_0 0))
.....
```

ぬりかべ

ニコリによる例題とルールの説明 ▸ Web

	3				1	
2			1			
	1			2		
		2				
1				1		6

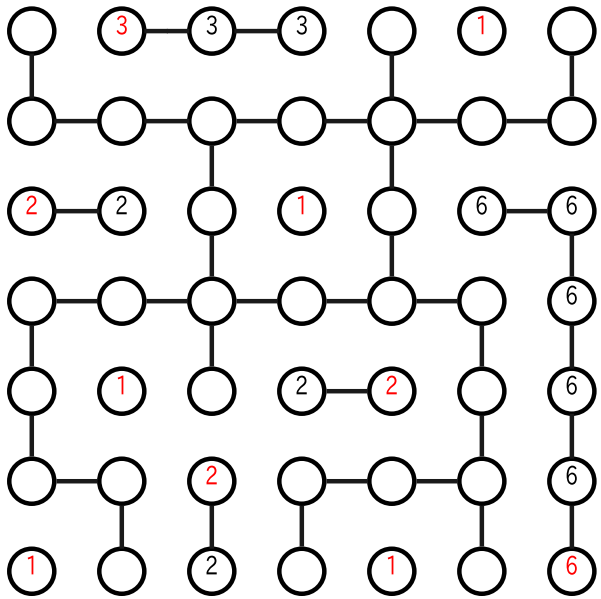


ぬりかべのルール

1. 以下のルールに従って盤面のマスをぬりつぶします．
2. 数字が入っているマスは黒マスにはなりません．
3. 数字は，その数字が含まれる，黒マスによって分断されたところ (シマとよぶ) のマスの数です．すべてのシマには数字が 1 つずつ入っていなければなりません．
4. すべての黒マスはタテヨコにひとつながりになっていなければなりません．
5. 黒マスが 2x2 マス以上のカタマリになってはいけません．

ぬりかべを Sugar で解く (1)

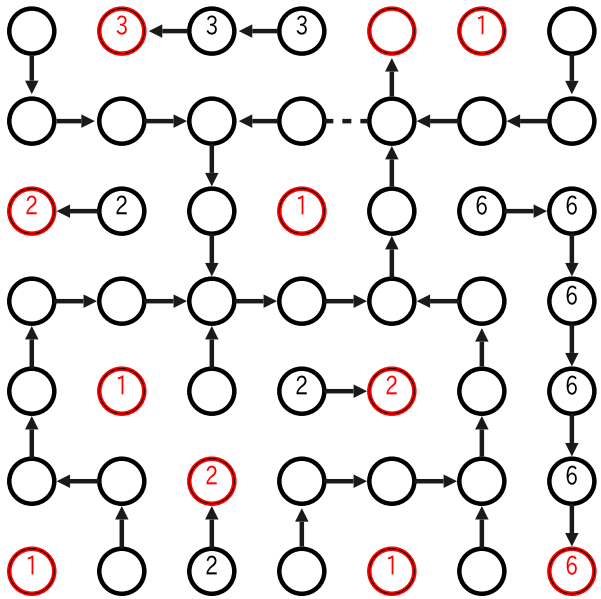
▸ Web



ぬりかべの制約記述

- 例題の場合，11 の領域に分かれる．

ぬりかべを Sugar で解く (1) ▸ Web



ぬりかべの制約記述

- 例題の場合，11 の領域に分かれる．
- ひとりにしてくれで述べたように，各連結成分のスパニング木を考え，根の個数が 11 になるように条件を記述する．
- 数字マスを含む領域については，数字マスが根になるようにする．

ぬりかべを Sugar で解く (2) ▸ Web

このままだと，白マス領域の大きさの条件が含まれていない．

ぬりかべの制約記述

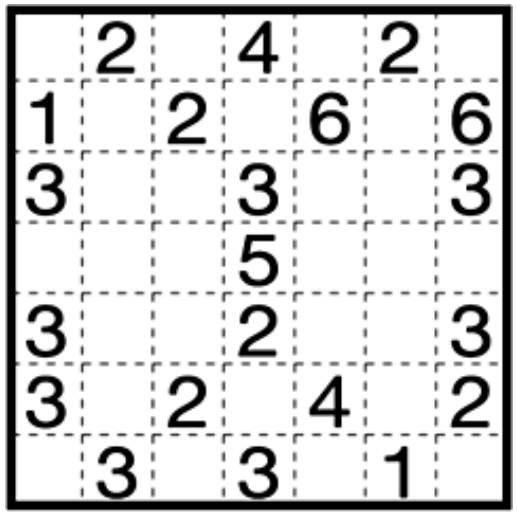
- 各白マスについて，自分および子孫のマスの総数を表わす変数 c_i_j を用意する．
- 数字マスについては，c_i_j の値はその数字に一致させる．
- 各白マスの c_i_j について条件を記述する．

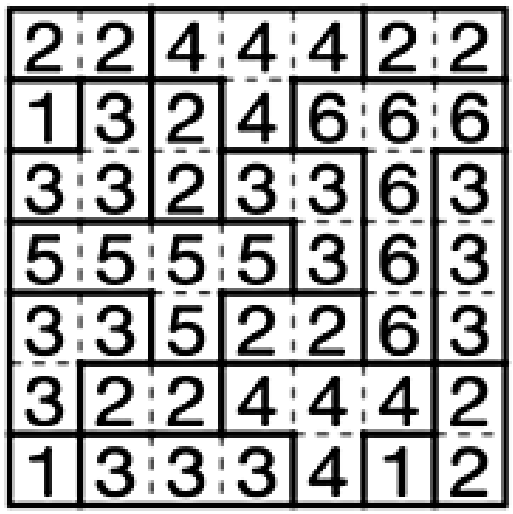
ぬりかべの記述例

```
(int c_0_0 1 6)
(int c_0_1 3 3)
.....
(=> (> x_0_0 0)
    (= c_0_0 (+ (if (< v_0_0 0) c_1_0 0)
                (if (< h_0_0 0) c_0_1 0) 1)))
.....
```

フィルオミノ

ニコリによる例題とルールの説明 ▸ Web





フィルオミノのルール

1. すべてのマスに数字を 1 つずつ入れます．
2. タテヨコにつながった同じ数字が入るマスを 1 つのブロックとし，全体をいくつかに分割します．
3. 1 つのブロックのマスの数は，そのブロックに入っている数字と同じでなければいけません．
4. 同じ数字のブロックは辺を共有してはいけません．

フィルオミノを Sugar で解く

▸ Web

- ぬりかべと同様に考えれば，複数の連結領域に分けることができる．また，連結領域の大きさも分かる．
- しかし，同じ大きさの領域が隣接しない，という条件が記述できない．

フィルオミノの制約記述

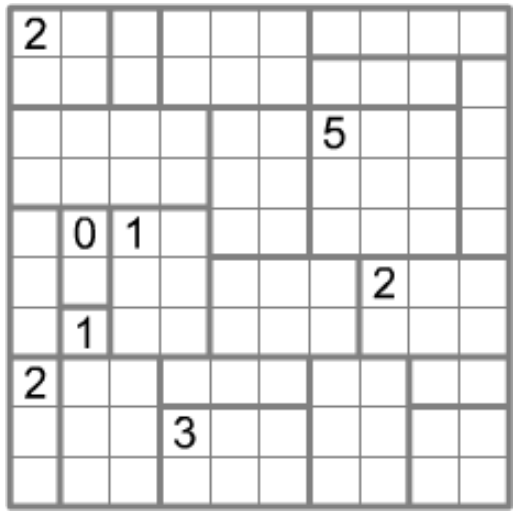
- 領域の番号を表わす変数 q_*i*_*j* を用意する．
- q_*i*_*j* の値は，根についてはその位置を表す番号に一致させ，根以外については親の値に一致させる．
- 隣接するマスについて，それぞれが属する領域の大きさが同じなら，領域番号が一致する制約条件を記述する．

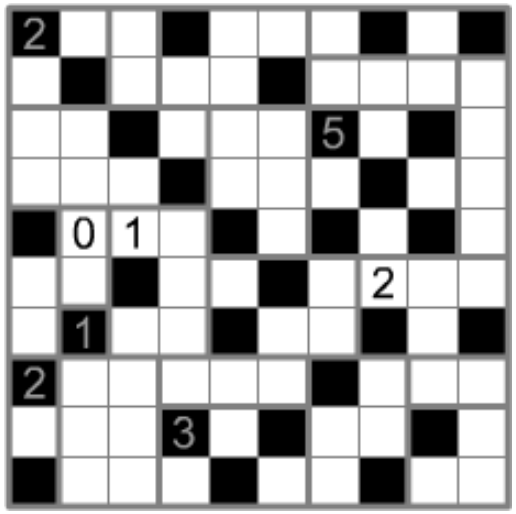
フィルオミノの記述例

```
(int q_0_0 1 49)
.....
(=> (> r_0_0 0) (= q_0_0 1))
(=> (> v_0_0 0) (= q_0_0 q_1_0))
.....
(=> (= x_0_0 x_1_0) (= q_0_0 q_1_0))
```

へやわけ

ニコリによる例題とルールの説明 ▸ Web





へやわけのルール

1. 以下のルールに従って，盤面に黒マスを配置します．
2. 出ている数字は，太線で区切られた四角（部屋）の中に入る黒マスの数です．数字の入っていない部屋には，いくつ黒マスが入るかわかりません．
3. 白マスは，タテまたはヨコにまっすぐに 3 つの部屋にわたって続いてはいけません．
4. 黒マスはタテヨコに連続しません．また，黒マスによって盤面が分断されることはありません．

へやわけを Sugar で解く

▸ Web

へやわけの制約記述

- i 行 j 列目を黒にするかどうかを $x\_i\_j \in \{0, 1\}$ で表す (0 が黒).
- 黒マスの数に関する条件を記述する.
- 白マスの連続に関する条件を記述する.
- 黒マスの連続に関する条件を記述する.
- 白マスの連結条件については，ひとりにしてくれと同様にする.

へやわけの記述例

```
(int x_0_0 0 1)
.....
(= (+ x_0_0 x_0_1 x_1_0 x_1_1) 2)
.....
(or (> x_0_1 0) (> x_0_2 0) (> x_0_3 0))
.....
(or (= x_0_0 0) (= x_1_0 0))
.....
```

まとめ

パズル雑誌のニコリ等にあるパズルを Sugar で解いてみた.

解いてみた感想

まとめ

パズル雑誌のニコリ等にあるパズルを Sugar で解いてみた．

解いてみた感想

- **ジャイアントサイズの難しい問題でも，結構解ける!**

まとめ

パズル雑誌のニコリ等にあるパズルを Sugar で解いてみた．

解いてみた感想

- ジャイアントサイズの難しい問題でも，結構解ける!
- 一方，なかなか解けない問題もあった．
  - ナンバーリンクは，15×15 の問題でも，今回の変換方法では解けなかった．工夫が必要．

まとめ

パズル雑誌のニコリ等にあるパズルを Sugar で解いてみた．

解いてみた感想

- ジャイアントサイズの難しい問題でも，結構解ける!
- 一方，なかなか解けない問題もあった．
  - ナンバーリンクは，15×15 の問題でも，今回の変換方法では解けなかった．工夫が必要．
- 今後は，以下を考えたい．
  - 制約記述の整理 (同様の制約記述がしばしば現れた)
  - 性能評価

まとめ

パズル雑誌のニコリ等にあるパズルを Sugar で解いてみた．

解いてみた感想

- ジャイアントサイズの難しい問題でも，結構解ける!
- 一方，なかなか解けない問題もあった．
  - ナンバーリンクは，15×15 の問題でも，今回の変換方法では解けなかった．工夫が必要．
- 今後は，以下を考えたい．
  - 制約記述の整理 (同様の制約記述がしばしば現れた)
  - 性能評価
- Web ページを見て，Sugar に興味を持ってくれる人もいた．

プログラム不要の「制約プログラミング手習い」 ▸ Web (by 藤原さん)