

Chapter 20: Working with Implications

Helmut Simonis

Cork Constraint Computation Centre
Computer Science Department
University College Cork
Ireland

ECLiPSe ELearning [Overview](#)



Licence

This work is licensed under the Creative Commons Attribution-Noncommercial-Share Alike 3.0 Unported License.

To view a copy of this license, visit [http:](http://creativecommons.org/licenses/by-nc-sa/3.0/)

[//creativecommons.org/licenses/by-nc-sa/3.0/](http://creativecommons.org/licenses/by-nc-sa/3.0/) or

send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.



Outline

- 1 Problem
- 2 Program
- 3 Improvements



What we want to introduce

- Solving a placement problem without specialized constraints
- Decomposition into pattern generation and set partitioning
- Using implications to propagate information



Problem Definition

Shikaku

The puzzle is played in a given recti-linear grid. Some grid cells contain numbers. The task is to partition the grid area into rectangular *rooms* satisfying the conditions:

- ① Each room contains exactly one number.
- ② The area of the room is equal to the number in it.

9			12		5		
							6
8	6	8					
				6	8	12	
4							
	3		9				4

9			12		5		
							6
8	6	8					
				6	8	12	
4							
	3		9				4



Solution Approaches

- The right way
- The wrong way



The Right Way (I)

- Using **one** `geost` constraint (Carlsson, Beldiceanu et al.)
- Each room is an object with fixed surface
- Shapes defined by $Width \times Height = Surface$
- Each room has x, y position and size w, h
- Each room must contain cell with hint
- Rooms must fit into given space
- Rooms do not overlap



The Right Way (II)

- Using `regular` (Lagerkvist, Pesant)
- 0/1 Variables x_{ijk} cell (i, j) belongs to room k
- Each cell must belong to exactly one room (equality)
- Use regular expression to describe possible shapes
- Disjunction of possible placements
- One `regular` constraint for each room
- Possible to combine regular expressions for multiple rooms



The Right Way (III)

- SMT?

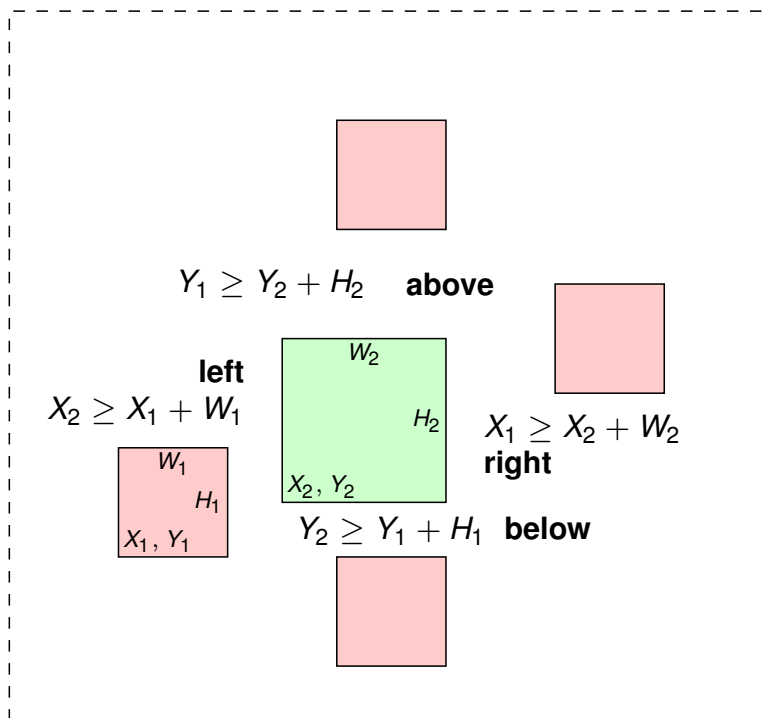


The Wrong Way (Naive)

- Each room k has position x_k, y_k and size w_k, h_k
- Room k has fixed surface $S_k = w_k * h_k$
- Room k contains fixed hint at U_k, V_k
 - $x_k \leq U_k < x_k + w_k$
 - $y_k \leq V_k < y_k + h_k$
- Any rooms k and l do not overlap



Expressing Non-Overlap with Disjunctive



Problems

- Weak propagation
- *Width* and *Height* not in sync
- Estimate for room surface much too low
- Disjunctive does not propagate much



- Problem decomposition
 - Phase 1: Generate possible placement pattern for each room
 - Phase 2: Pick exactly one pattern for each room



Phase 1: Pattern Generation

- Solve with finite domains
- For each room, solve
 - Each room k has position x_k, y_k and size w_k, h_k
 - Room k has fixed surface $S_k = w_k * h_k$
 - Room k contains fixed hint at U_k, V_k
 - $x_k \leq U_k < x_k + w_k$
 - $y_k \leq V_k < y_k + h_k$
 - All other hints are outside the room
 - Expressed as negation of inside
- Find all solutions



Phase 1 Results

- Possible placements for each room $1 \leq p \leq P_k$
- Position x_{kp} , y_{kp} , size w_{kp}, h_{kp}
- Predicate $q(k, p, i, j)$ pattern p of room k contains cell (i, j)



Phase 2: Set Partitioning

- 0/1 integer variable z_{kp} if pattern p is used for room k
- Select one pattern per room
- Cover every cell with exactly one pattern
- Constrain all variables which belong to pattern which contain cell



Model Phase 2

solve

$$z_{kp} \in \{0, 1\}$$

$$\forall k \in K : \sum_{1 \leq p \leq P_k} z_{kp} = 1$$

$$\forall (i, j) \in \text{Rect} : \sum_{\{k, p | q(k, p, i, j)\}} z_{kp} = 1$$

The first set of equations is subsumed by the second



Intuition Behind Constraints

- If only one pattern remains possible for a room, this must be chosen
- If a pattern is selected for a room, no other pattern can be selected
- Every cell must be covered by a pattern
- If a pattern covering some cell is selected, no other pattern covering the same cell may be selected



Top Level

```
:-module(pure) .
:-export(top/0) .
:-use_module(structures) .
:-use_module(data) .
:-use_module(utility) .
:-lib(ic) .

top:-
    data(Set,Nr,Grade,X,Y,Matrix) ,
    solve(Set,Nr,Grade,X,Y,Matrix) .
```



Structure Definitions

```
:-module(structures) .

:-export struct(hint(i,j,n,d)) .
:-export
struct(rectangle(x,y,w,h,
    n, % size of rectangle
    k, % hint nr, links alternative rectangles
    cnt, % id of rectangle
    var % 0/1 variable, design used
)) .
:-export struct(overlap(point,cnt,k,var)) .
```



Sample Data

```
:-module(data).
:-export(data/6).
data('nikoli-web' ,0 ,easy,10 ,10 ,
    [] (
        [] (x ,8 ,4 ,x ,x ,x ,x ,4 ,x ,x ),
        [] (x ,x ,x ,x ,x ,x ,x ,6 ,x ,x ),
        [] (x ,x ,x ,3 ,3 ,x ,x ,x ,x ,x ),
        [] (x ,x ,x ,x ,x ,x ,2 ,x ,x ,x ),
        [] (5 ,4 ,x ,x ,x ,x ,2 ,x ,x ,x ),
        [] (x ,x ,x ,9 ,x ,x ,x ,x ,6 ,7 ),
        [] (x ,x ,x ,8 ,x ,x ,x ,x ,x ,x ),
        [] (x ,x ,x ,x ,x ,4 ,5 ,x ,x ,x ),
        [] (x ,x ,4 ,x ,x ,x ,x ,x ,x ,x ),
        [] (x ,x ,5 ,x ,x ,x ,x ,5 ,6 ,x )
    )
).
```



Main Program (I)

```
solve(Set,Nr,Grade,X,Y,Matrix):-
    writeln(solving(Set,Nr,Grade)),
    (multifor([I,J],[1,1],[X,Y]),
        fromto([],A,A1,Hints),
        fromto(0,B,B1,_Types),
        param(Matrix) do
            hint_cell(Matrix,I,J,A,A1,B,B1)
        ),
    create_rectangles(Hints,[],[],
        Rectangles,X,Y),
    numbering(Rectangles),
    extract_vars(Rectangles,var of rectangle,List),
    List :: 0..1,
    ...
```



Main Program (II)

```
...
create_overlap(Rectangles, Overlap),
group_by(point of overlap, Overlap, Grouped),
(foreach(_-Group, Grouped) do
    extract_vars(Group, var of overlap, List),
    sum(List) #= 1
),
search(List, 0, input_order, indomain,
        complete, []),
writeln(List).
```



Extracting Hints

```
hint_cell(Matrix, I, J,
           A, [hint{i:I, j:J, n:N, d:D1} | A],
           D, D1) :-
    subscript(Matrix, [I, J], N),
    integer(N),
    !,
    D1 is D+1.
hint_cell(_Matrix, _I, _J, A, A, B, B).
```



Creating Pattern

```

create_rectangles([],_,Rectangles,Rectangles,
                 _Width,_Height).
create_rectangles([Hint|Hints],Old,RIn,ROut,
                 Width,Height):-
    findall(Rectangle,
            rectangle(Hint,Hints,Old,Width,Height,
                    Rectangle),
            Rectangles),
    append(Rectangles,RIn,R1),
    create_rectangles(Hints,[Hint|Old],R1,ROut,
                    Width,Height).

```



Phase 1 FD Model

```

rectangle(hint{i:I,j:J,n:N,d:K},Hints,Old,
         Width,Height,
         rectangle{x:X,y:Y,w:W,h:H,n:N,k:K}) :-
    X :: 1..Width,
    Y :: 1..Height,
    W :: 1..N,
    H :: 1..N,
    W*H #= N,
    X+W-1 #=< Width,
    Y+H-1 #=< Height,
    inside(X,Y,W,H,I,J),
    outsides(X,Y,W,H,Hints),
    outsides(X,Y,W,H,Old),
    search([X,Y,W,H],0,input_order,indomain,
          complete,[]).

```

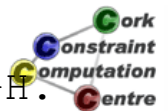


Phase 1 FD Model Utilities

```
inside(X, Y, W, H, I, J) :-
    I #>= X,
    J #>= Y,
    I #< X+W,
    J #< Y+H.
```

```
outsides(X, Y, W, H, L) :-
    (foreach(hint{i:I, j:J}, L),
     param(X, Y, W, H) do
        outside(X, Y, W, H, I, J)
    ).
```

```
outside(X, Y, W, H, I, J) :-
    I #< X or J #< Y or I #>=X+W or J #>= Y+H.
```



Creating Overlap Structures

```
create_overlap(Rectangles, Overlap) :-
    (foreach(Rect, Rectangles),
     fromto([], A, A1, Overlap) do
        create_overlap1(Rect, A, A1)
    ).
```

```
create_overlap1(rectangle{x:X, y:Y, w:W, h:H,
                        cnt:Cnt, k:K, var:Var}, In, Out) :-
    (multifor([I, J], [X, Y], [X+W-1, Y+H-1]),
     fromto(In, A,
            [overlap{point:point(I, J),
                    cnt:Cnt, k:K, var:Var}|A], Out),
     param(Cnt, Var, K) do
        true
    ).
```



Utility

```
extract_vars (Structures, Arg, List) :-
    (foreach (Struct, Structures),
     param (Arg),
     foreach (X, List) do
         arg (Arg, Struct, X)
    ).
```

```
numbering (Rectangles) :-
    (foreach (rectangle {cnt:C}, Rectangles),
     count (C, 1, _) do
         true
    ).
```



Result After Constraint Setup

1 ₄								29 ₆	
		9 ₆		16 ₃					33 ₂
			13 ₄				25 ₃		
		10 ₄						30 ₄	
						22 ₉			
							26 ₄		
2 ₉			14 ₄						
	5 ₆			17 ₈					
3 ₆					19 ₆			31 ₄	
	6 ₉			18 ₄					34 ₆
					20 ₄			32 ₄	
						23 ₆			35 ₂
		11 ₄							
			15 ₈						
	7 ₆							27 ₆	
		12 ₂				24 ₄			
4 ₄					21 ₃		28 ₆		
	8 ₄								36 ₆



Observation

- Only small part of problem filled in
- Missing information
- Some cells must belong to a room regardless of pattern used



Missing Information, Example

1 ₄						29	29	29 ₆	
		9 ₆		16 ₃					33 ₂
			13 ₄				25 ₃		
		10 ₄						30 ₄	
						22 ₉			
							26 ₄		
2 ₉			14 ₄						
	5 ₆			17 ₈					
3 ₆						19 ₆		31 ₄	
	6 ₉			18 ₄					34 ₆
						20 ₄		32 ₄	
							23 ₆		35 ₂
		11 ₄							
			15 ₈						
	7 ₆							27 ₆	
		12 ₂				24 ₄			
4 ₄						21 ₃	28 ₆		
	8 ₄								36 ₆



Adding View

- Variables which indicate which room some cell belongs to
- 0/1 integer variables w_{ijk} , whether cell (i, j) belongs to room k
- Most w_{ijk} entries are zero, as room k does not cover cell (i, j)
- Each cell must belong to a room
 - $\forall(i, j) : \sum_{k \in K} w_{ijk} = 1$



Channeling Constraints

- A cell belongs to a room, if one of the pattern covering the cell is selected
- Linear constraint
 - $\forall(i, j, k) : \sum_{\{p|q(k,p,i,j)\}} z_{kp} = w_{ijk}$
- Implications
 - $\forall(k, p, i, j) \text{ s.t. } q(k, p, i, j) : z_{kp} \Rightarrow w_{i,j,k}$
- Propagation is equivalent



Result After Improvement

1 ₄	9 ₁	9 ₁	29 ₁₃	29 ₁₆	29	29 ₂₂	29 ₂₅	29 ₆	33 ₂₉
9 ₁	9 ₁	9 ₆	16 ₁₃	16 ₃	16	29 ₂₂	29 ₂₅	33 ₃₀	33 ₂
13 ₉	13 ₁₀	13 ₁₀	13 ₄	22 ₁₆	25 ₂₂	25 ₂₂	25 ₁₃	30 ₂₅	33 ₃₀
10 ₁	10	10 ₄	14 ₁₃	22 ₁₆	30 ₂₂	30 ₂₂	30 ₂₆	30 ₂₅	30
2	2	2	14 ₁₃	22 ₁₇	22 ₁₉	22 ₁₇	22 ₉	30 ₂₆	34 ₃₀
2	2	2	14 ₁₃	26 ₂₂	26 ₂₂	26 ₁₉	26 ₂₂	31 ₃₀	34 ₂₈
2	2	2	14 ₄	22 ₁₇	22 ₁₉	26 ₂₃	26 ₁₇	31 ₃₀	34
3	5 ₆	5 ₅	5	17 ₈	19 ₁₇	23 ₂₂	31 ₂₈	31	34
3 ₆	5	5	5	18	19 ₆	31 ₂₃	31 ₂₆	31 ₄	34
3	6 ₉	6	6	18 ₄	20 ₁₉	23 ₂₂	32 ₃₁	32	34 ₆
3	6	6	6	18	20 ₄	32 ₂₃	32 ₂₇	32 ₄	35 ₃₄
3	6	6	6	18	23 ₆	32 ₂₇	35 ₃₂	35	35 ₂
3	11 ₄	11 ₄	15 ₁₁	15 ₁₁	23 ₂₀	27 ₂₄	27 ₂₃	32 ₂₇	36 ₃₅
15 ₇	15 ₁₁	15 ₁₁	15 ₈	15	27 ₂₃	27 ₂₄	27 ₂₃	32 ₂₇	36 ₂₇
7 ₄	7 ₆	27 ₁₅	27 ₁₅	27 ₁₅	27 ₂₄	27 ₂₄	27 ₁₅	27	36 ₂₇
7 ₄	12 ₇	12 ₂	24 ₁₅	24 ₁₅	24	28 ₂₇	28 ₂₄	36 ₂₈	36 ₂₈
4 ₄	8 ₄	12 ₈	21 ₁₅	21 ₁₅	21 ₃	28 ₂₄	28 ₂₁	36 ₂₈	36 ₂₈
8 ₄	8 ₄	8	8	36 ₈	36 ₂₁	36 ₂₈	36 ₂₈	36 ₂₈	36 ₆



Helmut Simonis

Shikaku

35

Not What We Expected

- Some cells are marked
- Others are still missing
- Cells marked can only be covered by one room
- Why is this happening?



Helmut Simonis

Shikaku

36

Missing Reasoning: Example

- $A, B, C \in \{0, 1\}$
- $A + B + C = 1 \wedge A + B = 1 \Rightarrow C = 0$
- This does not happen!
- Constraints only see variables, not other constraints
- We would need global constraint on sets of equations for this



Implications Too Weak

- We can add redundant constraints
- If a cell belongs to a room, no pattern for other rooms using that cell can be selected
 - $\forall(i, j, k) : \sum_{\{k', p' | q(k', p', i, j) \wedge k \neq k'\}} z_{k'p'} + w_{ijk} = 1$
 - $\forall(i, j, k, k', p') \text{ s.t. } q(k', p', i, j) : w_{ijk} \Rightarrow \neg z_{k'p'}$
- If a cell belongs to a room, no pattern for this room which does not cover the cell can be selected
 - $\forall(i, j, k) : \sum_{\{p | \neg q(k, p, i, j)\}} z_{kp} + w_{ijk} = 1$
 - $\forall(i, j, k, p) \text{ s.t. } \neg q(k, p, i, j) : w_{ijk} \Rightarrow \neg z_{kp}$



Result After Adding Redundant Constraints

1 ₄	9	9	9	29	29	29	29	29	29	29
1	9	9	9	16	16	16	25	33	33	33
1	10	10	13	13	13	13	25	30	30	30
1	10	10	14	22	22	22	25	30	30	30
2	2	2	14	22	22	22	26	26	34	34
2	2	2	14	22	22	22	26	26	34	34
2	2	2	14	17	17	17	17	31	34	34
3	5	5	5	17	17	17	17	31	34	34
3	5	5	5	18	19	19	19	31	34	34
3	6	6	6	18	19	19	19	31	34	34
3	6	6	6	18	20	32	32	32	32	32
3	6	6	6	18	20	23	23	23	35	35
3	11	11	11	11	20	23	23	23	35	35
7	7	7	15	15	20	24	27	27	27	27
7	7	7	15	15	21	24	27	27	27	27
4	4	12	15	15	21	24	28	28	28	28
4	4	12	15	15	21	24	28	28	28	28
8	8	8	8	36	36	36	36	36	36	36



Problem Solved?

- More to be done
- Look through example problems
- Find cases where problem is not solved by propagation
- Try and find redundant constraints



Space required for rooms 76, 94, 102, 107

1	30	30	30	30	30	30	30	30	43	43	43	43	56	60	66	66	74	74	74	74	74	103	103	103	108	116	116	116	116	116	116	139	147	147							
1	8	8	12	18	18	18	18	35	43	43	43	43	56	60	66	66	91	91	91	91	91	91	99	99	108	112	129	129	129	129	133	139	147	147							
1	8	8	12	22	22	26	26	35	39	39	48	52	56	60	66	66	77	77	85	85	85	85	99	99	108	112	117	121	125	130	133	139	147	147							
1	8	8	12	22	22	26	26	35	39	39	48	52	56	60	66	66	77	77	81	81	81	95	99	99	108	112	117	121	125	130	133	139	143	143							
1	4	4	13	22	22	27	31	35	39	39	48	52	56	70	70	70	77	77	86	86	86	95	99	99	108	112	117	121	125	130	134	134	143	143							
1	5	9	13	19	23	27	31	35	40	44	48	52	56	61	71	71	71	78	86	86	86	95	104	108	113	117	121	125	130	134	134	144	144								
1	5	9	14	19	23	27	31	35	40	44	48	52	57	61	71	71	71	78	86	86	86	96	104	109	113	117	121	126	130	134	134	144	144								
1	5	9	14	19	23	27	32	36	40	44	48	53	57	61	67	75	75	78	82	82	92	96	100	109	113	117	121	126	130	140	140	140	148								
1	5	9	14	19	24	27	32	36	40	49	49	53	57	62	67	75	75	78	87	87	92	96	100	109	113	117	122	126	130	135	135	145	148								
2	5	15	15	19	24	28	32	37	40	45	45	53	57	62	67	75	75	78	87	87	92	96	100	109	113	117	122	126	130	135	135	145	148								
2	5	15	15	19	24	28	32	37	40	45	45	53	58	58	67	75	75	78	87	87	92	96	105	109	118	118	122	126	130	135	135	145	148								
2	5	15	15	20	24	28	32	37	41	41	50	53	58	58	67	75	75	78	87	87	93	96	105	109	118	118	122	131	131	135	135	145	148								
2	5	15	15	20	24	28	32	37	41	41	50	53	63	63	67	88	88	88	88	93	96	101	101	101	101	123	123	123	136	136	141	145	148								
2	10	10	10	10	24	33	33	37	41	41	50	54	63	63	68	68	68	68	83	83	93	97	97	97	110	114	114	114	127	127	127	141	145	148							
6	6	11	16	16	24	33	33	37	42	46	50	54	63	63	72	72	72	79	79	79	93	97	97	97	110	114	114	114	127	127	127	141	145	148							
7	7	11	16	16	24	33	33	38	42	46	50	54	64	64	64	64	64	64	89	89	89	89	106	106	106	119	119	119	119	137	141	145	149								
7	7	11	25	25	25	25	25	38	42	46	50	73	73	73	73	73	73	80	80	80	80	80	80	80	80	80	80	80	80	80	80	124	128	128	137	141	146	149			
7	7	11	25	25	25	25	25	38	42	46	50	73	73	73	73	73	73	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	124	128	128	137	141	146	149
7	7	11	25	25	25	25	25	38	42	46	50	55	55	55	55	55	69	69	69	69	84	84	84	98	98	98	111	115	115	124	128	128	142	142	146	149					
17	17	17	17	17	17	29	29	29	29	51	51	51	51	65	65	65	65	90	90	90	90	102	102	111	115	115	124	138	138	138	138	146	149								
		21	21	21	21								59	65	65	65	65	90	90	90	94	94				132	132	132	132	132	132	146	149								
3	3					34			47	47	47	47					76	76					107	107	107	107	120	120	120	120	120	120	120	149							



How to model this?

- Another view: finite domain variables for room assignment
- Variables v_{ij} state that cell (i, j) belongs to room v_{ij}
- Connection to w_{ijk} variables via $\text{bool_channel}(v_{ij}, [w_{ij1}, w_{ij2}, \dots, w_{ij|K|}])$ constraints
- Each value must occur specified time
- gcc constraint with fixed counts



Improvement

- gcc reasons on all hints together
- Just adding $\sum_{(i,j)} w_{ijk} = S_k$ does not do this



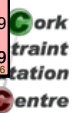
Deduction by gcc (partial)

1	30	30	30	30	30	30	30	30	43	43	43	43	56	60	66	66	74	74	74	74	74	103	103	103	108	116	116	116	116	116	116	139	147	147		
1	8	8	12	18	18	18	18	35	43	43	43	43	56	60	66	66	91	91	91	91	91	91	99	99	108	112	129	129	129	129	133	139	147	147		
1	8	8	12	22	22	26	26	35	39	39	48	52	56	60	66	66	77	77	85	85	85	85	99	99	108	112	117	121	125	130	133	139	143	143		
1	8	8	12	22	22	26	26	35	39	39	48	52	56	60	66	66	77	77	81	81	81	95	99	99	108	112	117	121	125	130	134	134	143	143		
1	4	4	13	22	22	27	31	35	39	39	48	52	56	70	70	70	77	77	86	86	86	95	99	99	108	112	117	121	125	130	134	134	143	143		
1	5	9	13	19	23	27	31	35	40	44	48	52	56	61	71	71	71	78	86	86	86	95	104	108	113	117	121	125	130	134	134	144	144			
1	5	9	14	19	23	27	31	35	40	44	48	52	57	61	71	71	71	78	86	86	86	96	104	109	113	117	121	126	130	134	134	144	144			
1	5	9	14	19	23	27	32	36	40	44	48	53	57	61	67	75	75	78	82	82	92	96	100	109	113	117	121	126	130	140	140	140	148			
1	5	9	14	19	24	27	32	36	40	49	49	53	57	62	67	75	75	78	87	87	92	96	100	109	113	117	122	126	130	135	135	145	148			
2	5	15	15	19	24	28	32	37	40	45	45	53	57	62	67	75	75	78	87	87	92	96	100	109	113	117	122	126	130	135	135	145	148			
2	5	15	15	19	24	28	32	37	40	45	45	53	58	58	67	75	75	78	87	87	92	96	105	109	118	118	122	126	130	135	135	145	148			
2	5	15	15	20	24	28	32	37	41	41	50	53	58	58	67	75	75	78	87	87	93	96	105	109	118	118	122	131	131	135	135	145	148			
2	5	15	15	20	24	28	32	37	41	41	50	53	63	63	67	88	88	88	88	93	96	101	101	101	101	123	123	123	136	136	141	145	148			
2	10	10	10	10	24	33	33	37	41	41	50	54	63	63	68	68	68	83	83	93	97	97	97	110	114	114	114	127	127	127	141	145	148			
6	6	11	16	16	24	33	33	37	42	46	50	54	63	63	72	72	72	79	79	79	93	97	97	97	110	114	114	114	127	127	127	141	145	148		
7	7	11	16	16	24	33	33	38	42	46	50	54	64	64	64	64	64	89	89	89	89	106	106	106	119	119	119	119	137	141	145	149				
7	7	11	25	25	25	25	25	38	42	46	50	73	73	73	73	73	80	80	80	80	80	80	80	80	80	80	80	80	124	128	128	137	141	146	149	
7	7	11	25	25	25	25	25	38	42	55	55	55	55	55	69	69	69	69	84	84	84	98	98	98	111	115	115	124	128	128	142	142	146	149		
17	17	17	17	17	17	29	29	29	29	51	51	51	51	65	65	65	65	90	90	90	94	94	102	102	111	115	115	124	138	138	138	138	146	149		
		21	21	21	21									59	65	65	65	65	90	90	90	94	94				132	132	132	132	132	146	149			
3	3							34	47	47	47	47		76	76	76	76	76																		



Result

1	30	30	30	30	30	30	30	30	30	43	43	43	43	56	60	66	66	74	74	74	74	74	103	103	103	108	116	116	116	116	116	116	139	147	147					
1	8	8	12	18	18	18	18	35	35	43	43	43	43	56	60	66	66	91	91	91	91	91	91	99	99	108	112	129	129	129	129	133	139	147	147					
1	8	8	12	22	22	26	26	35	35	39	39	48	52	56	60	66	66	77	77	85	85	85	85	99	99	108	112	117	121	125	130	133	139	147	147					
1	8	8	12	22	22	26	26	35	35	39	39	48	52	56	60	66	66	77	77	81	81	81	95	99	99	108	112	117	121	125	130	134	134	143	143					
1	4	4	13	22	22	27	31	35	35	39	39	48	52	56	70	70	70	77	77	86	86	86	95	99	99	108	112	117	121	125	130	134	134	143	143					
1	5	9	13	19	23	27	31	35	35	40	44	48	52	56	61	71	71	71	78	86	86	86	95	104	104	108	113	117	121	125	130	134	134	144	144					
1	5	9	14	19	23	27	31	35	35	40	44	48	52	57	61	71	71	71	78	86	86	86	96	104	104	109	113	117	121	126	130	134	134	144	144					
1	5	9	14	19	23	27	32	36	36	40	44	48	53	57	61	67	75	75	78	82	82	92	96	100	104	109	113	117	121	126	130	140	140	140	148					
1	5	9	14	19	24	27	32	36	36	40	49	49	53	57	62	67	75	75	78	87	87	92	96	100	104	109	113	117	122	126	130	135	135	145	148					
2	5	15	15	19	24	28	32	37	37	40	45	45	53	57	62	67	75	75	78	87	87	92	96	100	105	109	113	117	122	126	130	135	135	145	148					
2	5	15	15	19	24	28	32	37	37	40	45	45	53	58	58	67	75	75	78	87	87	92	96	105	105	109	118	118	122	126	130	135	135	145	148					
2	5	15	15	20	24	28	32	37	37	41	41	50	53	58	58	67	75	75	78	87	87	93	96	105	105	109	118	118	122	131	131	135	135	145	148					
2	5	15	15	20	24	28	32	37	37	41	41	50	53	63	63	67	88	88	88	88	93	96	101	101	101	101	123	123	123	136	136	141	145	148						
2	10	10	10	10	24	33	33	37	37	41	41	50	54	63	63	68	68	68	68	83	83	93	97	97	97	110	114	114	114	127	127	127	141	145	148					
6	6	11	16	16	24	33	33	37	37	42	46	50	54	63	63	72	72	72	79	79	79	93	97	97	97	110	114	114	114	127	127	127	141	145	148					
7	7	11	16	16	24	33	33	38	38	42	46	50	54	64	64	64	64	64	64	89	89	89	89	106	106	106	119	119	119	119	119	119	137	141	145	149				
7	7	11	25	25	25	25	25	38	38	42	46	50	73	73	73	73	73	73	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	124	128	128	137	141	146	149
7	7	11	25	25	25	25	25	38	38	42	44	55	55	55	55	55	69	69	69	69	84	84	84	98	98	98	111	115	115	124	128	128	142	142	146	149				
17	17	17	17	17	17	29	29	29	29	51	51	51	51	65	65	65	65	90	90	90	90	102	102	102	102	111	115	115	124	138	138	138	138	146	149					
3	3	21	21	21	21	21	21	34	34	47	47	47	47	59	65	65	65	65	90	90	90	90	94	94	94	94	94	94	132	132	132	132	132	132	146	149				
3	3	21	21	21	21	21	21	34	34	47	47	47	47	59	76	76	76	76	76	107	107	107	107	107	107	107	107	120	120	120	120	120	120	120	120	149				



Unsolved Part

6	86	86	95	99	99	108	112	117	12	
6	86	86	95	104	104	108	113	117	12	
6	86	86	96	104	104	109	113	117	12	
2	82	92	96	100	104	109	113	117	12	
7	87	92	96	100	105	104	109	113	117	12
7	87	92	96	100	105	100	109	113	117	12
7	87	92	96	105	100	105	109	118	118	12
7	87	93	96	105	100	105	109	118	118	12
8	88	93	96	101	101	101	101	123	12	



What to do?

- Pattern are pair-wise compatible with each other
- There is enough space
- But there is only one solution



Looking more closely

6	86	86	95	99	99	108	112	117	126	86	86	95	99	99	108	112	117	126	8
6	86	86	95	104	104	108	113	117	126	86	86	95	104	104	108	113	117	126	8
6	86	86	96	104	104	109	113	117	126	86	86	96	104	104	109	113	117	126	8
2	82	92	96	100	104	109	113	117	122	82	92	96	100	104	109	113	117	122	8
7	87	92	96	100	104	109	113	117	127	87	92	96	100	104	109	113	117	127	8
7	87	92	96	100	105	109	113	117	127	87	92	96	100	105	109	113	117	127	8
7	87	92	96	105	105	109	118	118	127	87	92	96	105	105	109	118	118	127	8
7	87	93	96	105	105	109	118	118	127	87	93	96	105	105	109	118	118	127	8
8	88	93	96	101	101	101	101	123	128	88	93	96	101	101	101	101	123	128	8



As Equations

- For one cell we have
 - $a_{104} + c_{100} = 1$
- For the other cell we have
 - $a_{104} + c_{100} + a_{105} = 1$
 - Implies $a_{105} = 0$
 - Similar, $a_{104} = 0$
 - Therefore $c_{100} = 1$

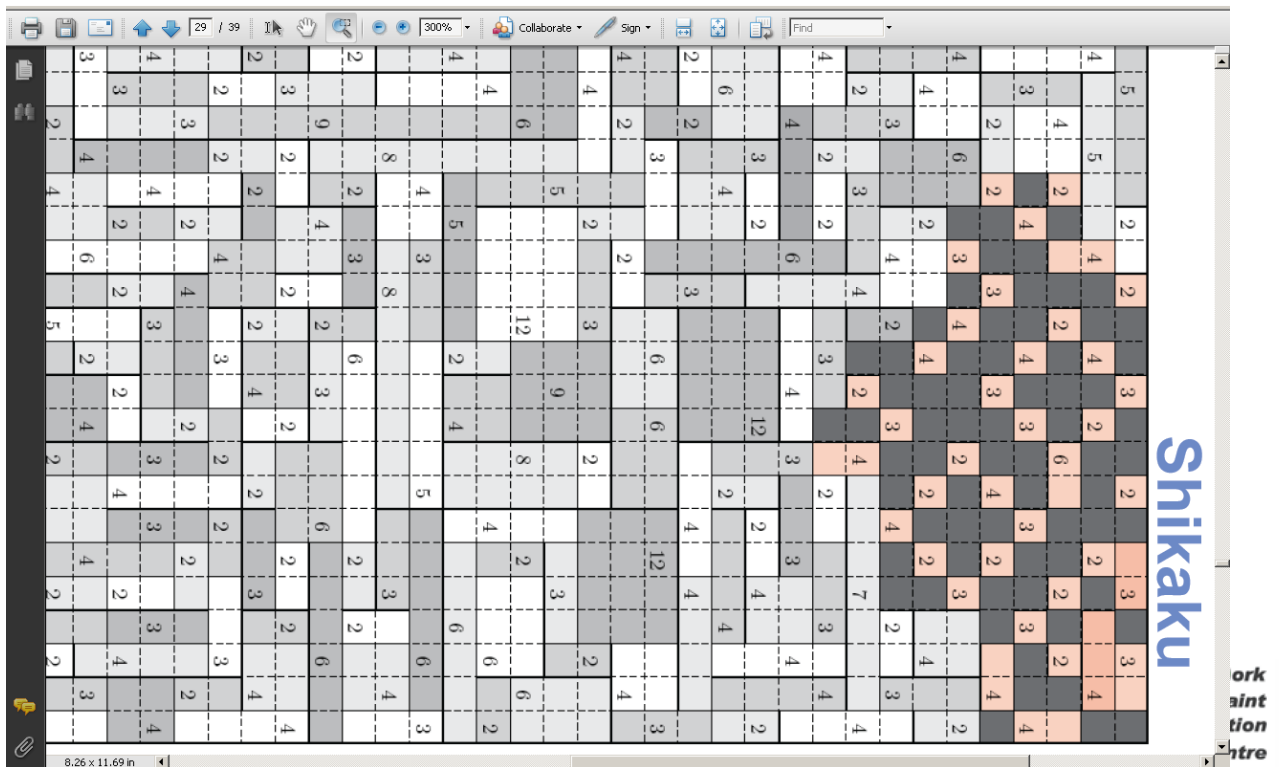


All done? No, One Problem Still Open

13	116 ₃	130	130	130 ₃	136	143	143	170	170	170	176	190	190	197 ₅	203
04	116	127	127 ₄	131	136	144	150 ₃	170	170	170	177 ₂	191 ¹⁸⁴ ₁₇₇	191 ₂	197	203
08	116	127	127	131 ₂	136	144	150	164	164 ₂	164 ₂	171 ¹⁸⁴ ₁₇₇	184 ₄	191 ₁₈₄	197	204 ₂
14	137	137	137	137	137	137	150	157	157	171 ₃	171 ₃	184 ¹⁹⁸ ₁₇₈	184 ¹⁹⁸ ₁₇₈	198	204
14	123	123 ₃	123	151	151	151	151 ₄	157	157	171 ₃	178 ₃	184 ¹⁹⁸ ₁₇₈	184 ¹⁹⁸ ₁₇₈	198	205 ₂
17	117	132	132	132	138	145	158	158 ₂	172 ₄	172 ₄	184 ¹⁹⁸ ₁₇₈	184 ¹⁹⁸ ₁₇₈	192 ₂	199	206 ²⁰⁵ ₁₉₉
17	117	132	132	132	138	145 ₃	165 ₁₅₂	165 ₁₅₂	165 ₁₅₂	165 ₁₅₂	179 ¹⁸⁸ ₁₆₅	179 ¹⁸⁸ ₁₆₅	185 ₄	199 ₄	206 ₁₉₉
17	117	132	132	132	138	145	152 ₂	165 ₁₅₂	165 ₁₅₂	165 ₁₅₂	179 ¹⁸⁸ ₁₆₅	179 ¹⁸⁸ ₁₆₅	185 ₄	199 ₄	206 ₃
18	118	132	132	132	138	145	159 ₁₅₃	159 ₁₅₃	159 ₁₅₃	159 ₁₅₃	179 ¹⁸⁸ ₁₆₅	179 ¹⁸⁸ ₁₆₅	186 ₃	200 ₂	206
18	118	124	139	139	139 ₃	153	153 ₄	153 ₄	153 ₄	153 ₄	179 ¹⁸⁸ ₁₆₅	179 ¹⁸⁸ ₁₆₅	186 ₃	200 ₂	206
18	118	124	128	128	140	146	154	166 ₂	166 ₂	166 ₂	180 ₄	180 ₄	193	207	207 ₂
19	119	124	129	133	140	146	154	160 ₄	160 ₄	160 ₄	178 ¹⁸⁷ ₁₆₀	178 ¹⁸⁷ ₁₆₀	187 ₃	208	207
19	119	124	129	133	140	147	154	167	167	167	178 ¹⁸⁷ ₁₆₀	178 ¹⁸⁷ ₁₆₀	181 ₂	201 ₂	208
19	119	125	129	134	134	147	154	167	167	167	178 ¹⁸⁷ ₁₆₀	178 ¹⁸⁷ ₁₆₀	181 ₂	201 ₂	208
19	119	125	129	134	134	147	154	161 ₂	168	168	178 ¹⁸⁷ ₁₆₀	178 ¹⁸⁷ ₁₆₀	188 ₃	202	208
15	115	125	141	141	141	141	154	161	168	168	182	182	195 ₂	202	209 ₃
15	115	125	148	148	148	148	154	162	162	162	182	182	195 ₂	202	209
20	120	120	135	135 ₂	155	155	155 ₄	155	155	175	175 ₂	189	189	202	209



Also Open with `geost` (Image: N. Beldiceanu)



Results (I)

Set	Nr	Grade	X	Y	K	Fix	Alt	Cov	Ini	Set	Red	GCC	Sub	SAC
nikoli-web	0	easy	10	10	20	3	80	6	0	0	0	0	0	0
nikoli-web	1	easy	10	10	14	2	81	5	0	0	0	0	0	0
nikoli-web	2	easy	10	10	16	1	83	2	0	0	0	0	0	0
nikoli-web	3	easy	10	10	28	3	96	12	0	0	0	0	0	0
nikoli-web	4	easy	10	18	44	15	78	78	0	0	0	0	0	0
nikoli-web	5	medium	10	18	38	3	115	21	0	0	0	0	0	0
nikoli-web	6	medium	10	18	36	1	210	3	153	50	0	0	0	0
nikoli-web	7	medium	14	24	68	4	293	16	0	0	0	0	0	0
nikoli-web	8	hard	14	24	70	8	380	17	0	0	0	0	0	0
nikoli-web	9	hard	20	36	128	4	872	8	196	0	0	0	0	0
nikoli-web	10	hard	20	36	120	1	986	1	774	627	36	36	0	0
giant	0	easy	7	7	12	2	46	2	0	0	0	0	0	0
giant	1	-	35	21	149	0	1002	2	995	345	29	7	0	0
giant	2	-	35	21	189	0	888	5	855	324	0	0	0	0
giant	3	-	35	21	209	5	1023	6	830	583	256	256	133	0
giant	4	-	35	21	190	2	849	8	495	70	0	0	0	0
giant	5	-	35	21	137	1	990	3	929	657	0	0	0	0
giant	6	-	35	21	160	2	900	5	323	113	0	0	0	0
giant	7	-	35	21	135	4	1111	4	984	866	7	7	0	0
giant	8	-	35	21	140	1	979	3	943	646	0	0	0	0

Results (II)

Set	Nr	Grade	X	Y	K	Fix	Alt	Cov	Ini	Set	Red	GCC	Sub	SAC
giant	1701	hard	45	31	255	10	1731	25	617	178	0	0	0	0
giant	1702	hard	45	31	206	3	1727	11	1497	277	0	0	0	0
giant	1801	hard	45	31	189	12	1468	35	0	0	0	0	0	0
giant	1802	hard	45	31	276	11	1517	34	769	217	0	0	0	0
giant	1901	hard	45	31	188	8	1443	36	148	18	0	0	0	0
giant	1902	hard	45	31	213	4	1813	11	1449	1018	0	0	0	0
giant	2001	hard	45	31	221	16	1217	92	0	0	0	0	0	0
giant	2002	hard	45	31	278	6	1617	20	1259	758	0	0	0	0



Why Model the Wrong Way?

- We want to mimic human reasoning
- Human don't have a built-in geost constraint
- Most humans use rules to describe solution process
 - If a cell can only be covered by one room, then it must be assigned to the room
- Both views (decide between pattern and assign cell to room) are used by humans
- At some point people
 - Either invent special global constraints
 - Or use SAC



Conclusions

- Shikaku - interesting little puzzle
- Solved by decomposition and reasoning on implications
- Could be solved by strong global constraint
- We use incremental process adding constraints as required
- Close to human solving method
- One open problem left, same for `geost` (M. Carlsson)

