```
-- q0 mark symbol
   d(q0,a) = (q1,#,R)
   d(q0,b) = (q3,#,R)
   d(q0,#) = (YES,#,L)

-- q1 move right, remember 'a'
   d(q1,a) = (q1,a,R)
   d(q1,b) = (q1,b,R)
   d(q1,#) = (q2,#,L)

-- q2 test if there is an 'a'
   d(q2,a) = (q5,#,L)
   d(q2,b) = (NO,#,L)
   d(q2,#) = (YES,#,L)

-- q3 move right, remember 'b'
   d(q3,a) = (q3,a,R)
   d(q3,b) = (q3,b,R)
   d(q3,#) = (q4,#,L)

-- q4 check if there is an 'b'
   d(q4,a) = (NO,#,L)
   d(q4,b) = (q5,#,L)
   d(q4,#) = (YES,#,L)

-- q5 return to leftmost nonblank
   d(q5,a) = (q5,a,L)
   d(q5,b) = (q5,b,L)
   d(q5,#) = (q0,#,R)
```

- Turing machine fourtuple (Q, Σ, δ, q_0)
  - Q states, Σ symbols, q_0 start state
  - δ transition function

    $\delta\colon Q \times \Sigma \to (Q \cup \{q_{\mathsf{halt}}, q_{\mathsf{yes}}, q_{\mathsf{no}}\}) \times \Sigma \times \{\mathtt{L}, \mathtt{R}\}$

- instanteneous description $xqay$
  - xay part of tape from leftmost to rightmost non-blank
  - tapehead at symbol a
  - machine in state q

- one-step derivation

  $xqay \vdash_M xbq'y$ if $\delta(q,a) = (q', b, R)$

  $xcqay \vdash_M xq'cby$ if $\delta(q,a) = (q', b, L)$

- the halting problem for Turing machines

  string m for TM M, input w for M

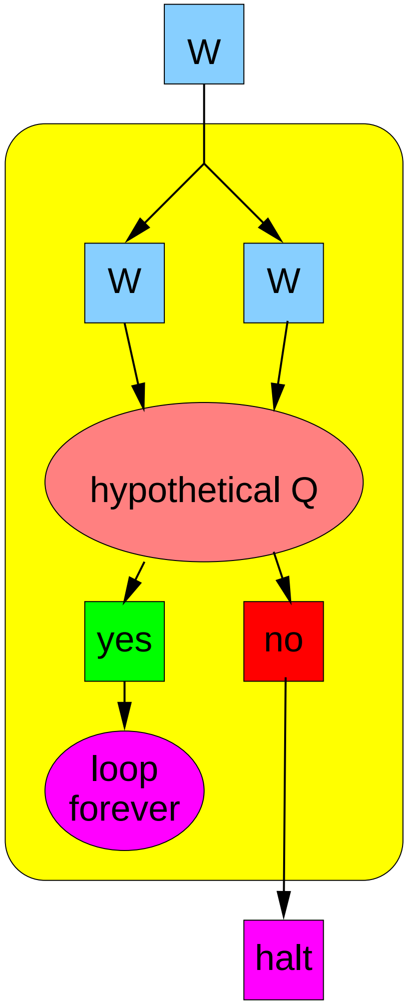
  does a TM HP exist such that

  $q_0mw \vdash^*_{HP} yq_{\text{yes}}z$ if TM M halts on input w

  $q_0mw \vdash^*_{HP} yq_{\text{no}}z$ if TM M does not halt on input w

- theorem the halting problem is undecidable

- the program S



- the state-entry problem

TM M, state q of M, non-empty x for M

does a decision algorithm $\mathcal{A}$ exist such that

$$\mathcal{A}(M, q, x) \iff \exists y, z \colon q_0 x \vdash_M^* yqz$$

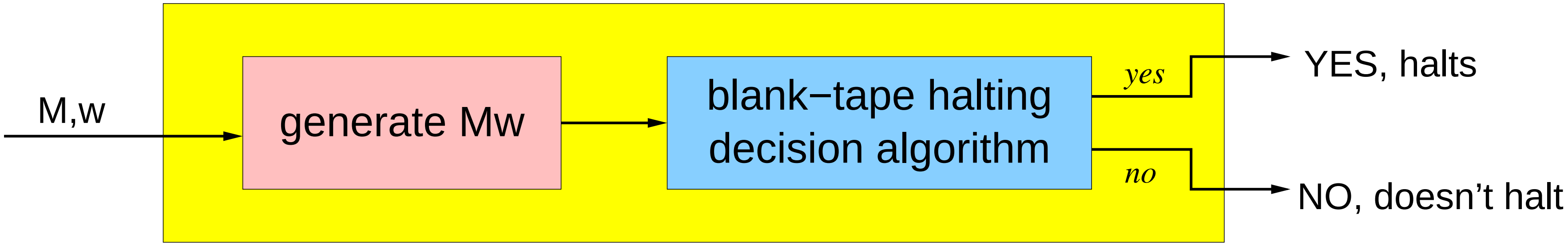
M' has $\delta(q_i, a) = (q_{\mathtt{SPEC}}, a, R)$ if $\delta(q_i, a)$ is undefined for M

M halts on input w iff $\mathcal{A}(M', q_{\mathtt{SPEC}}, w)$

- the blank-tape halting problem

  does a decision algorithm $\mathcal{B}$ exist such that

$$\mathcal{B}(M) \iff \exists y, z \colon q_0 \vdash_M^* y q_{\text{yes}} z$$



- non-emptiness $\mathcal{L}(M) = \emptyset$ for FA M is decidable
- equivalence $\mathcal{L}(M_1) = \mathcal{L}(M_2)$ for FAs M_1, M_2 is decidable

- non-emptiness $\mathcal{L}(M) = \emptyset$ for TM M is undecidable
- equivalence $\mathcal{L}(M_1) = \mathcal{L}(M_2)$ for TMs M_1, M_2 is undecidable

- word correspondence problem

	1	2	3	4	5
X	abb	a	bab	$baba$	aba
Y	$bbab$	aa	ab	aa	a

correspondence: 2, 1, 1, 4, 1, 5

	1	2	3	4	5
X	bb	a	bab	$baba$	aba
Y	bab	aa	ab	aa	a

admits no correspondence

- word correspondence problem

  $A = w_1, w_2, \ldots, w_n,\ \ B = v_1, v_2, \ldots, v_n$

  does a sequence $i_1, i_2, \ldots, i_k$ exist such that

  $$w_{i_1} w_{i_2} \cdots w_{i_k} = v_{i_1} v_{i_2} \cdots v_{i_k}$$

- theorem ambiguity of context-free grammars is undecidable

  for PCP $A = (w_1, \ldots, w_n)$ and $B = (v_1, \ldots, v_n)$ construct

  $S \to S_A \mid S_B$

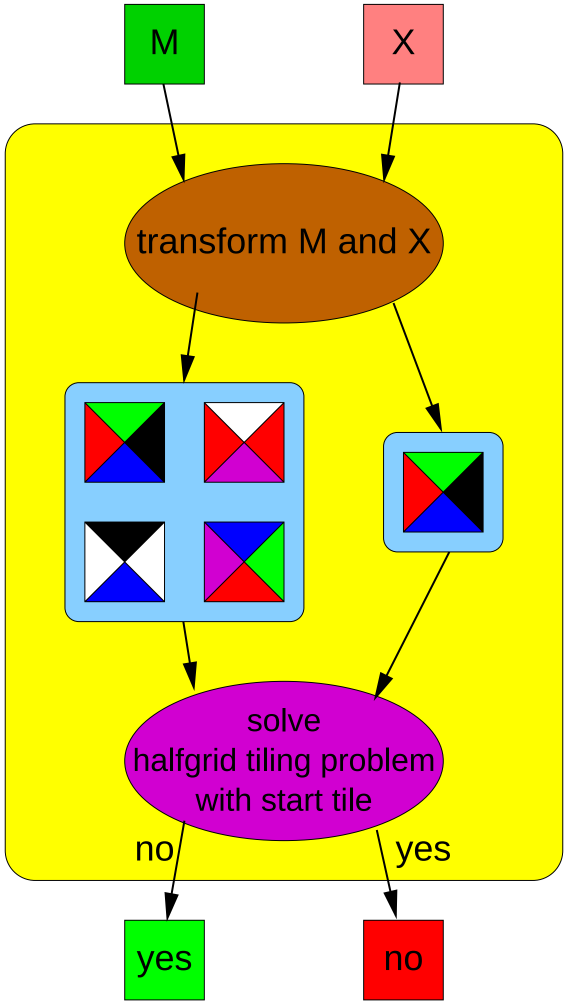
  $S_A \to w_i S_A a_i \mid w_i a_i$

  $S_B \to v_i S_B a_i \mid v_i a_i$

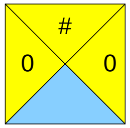
  if G is ambigious then $w_{i_1} \ldots w_{i_k} = v_{i_1} \ldots v_{i_k}$

- theorem the non-empty intersection problem of context-free grammars is undecidable

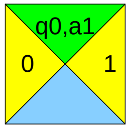
- half-plane tiling with start tile



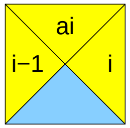
- basic tiles



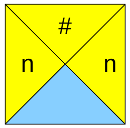
padding left



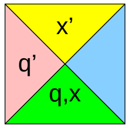
start state $q0$, head at $a1$



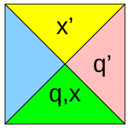
symbol ai at position i



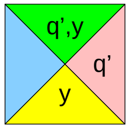
padding right



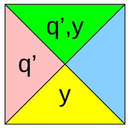
$\delta(q, x) = (q', x', L)$



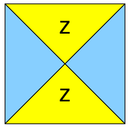
$\delta(q, x) = (q', x', R)$



head moves left into state q'

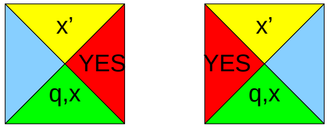


head moves right into state q'

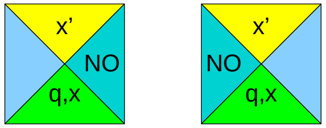


cell remains unchanged

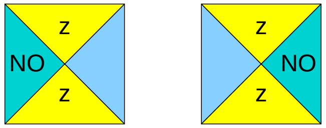
- special tiles



acceptance

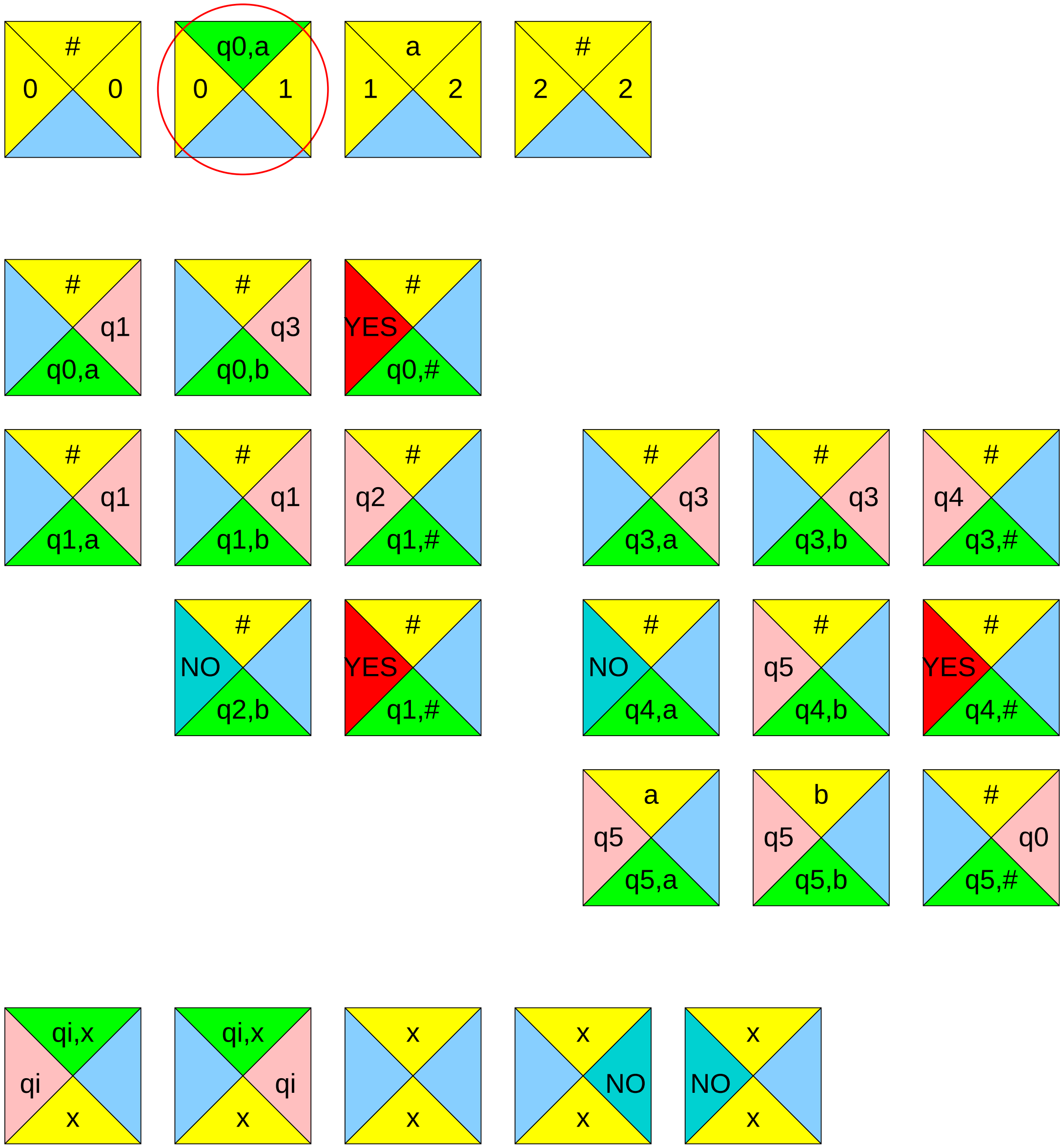


rejection



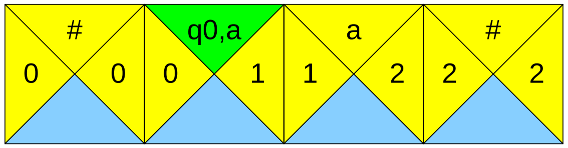
rejection pad

```
-- q0 mark symbol
   d(q0,a) = (q1,#,R)
   d(q0,b) = (q3,#,R)
   d(q0,#) = (YES,#,L)

-- q1 move right, remember 'a'
   d(q1,a) = (q1,a,R)
   d(q1,b) = (q1,b,R)
   d(q1,#) = (q2,#,L)

-- q2 test if there is an 'a'
   d(q2,a) = (q5,#,L)
   d(q2,b) = (NO,#,L)
   d(q2,#) = (YES,#,L)

-- q3 move right, remember 'b'
   d(q3,a) = (q3,a,R)
   d(q3,b) = (q3,b,R)
   d(q3,#) = (q4,#,L)

-- q4 check if there is an 'b'
   d(q4,a) = (NO,#,L)
   d(q4,b) = (q5,#,L)
   d(q4,#) = (YES,#,L)

-- q5 return to leftmost nonblank
   d(q5,a) = (q5,a,L)
   d(q5,b) = (q5,b,L)
   d(q5,#) = (q0,#,R)
```

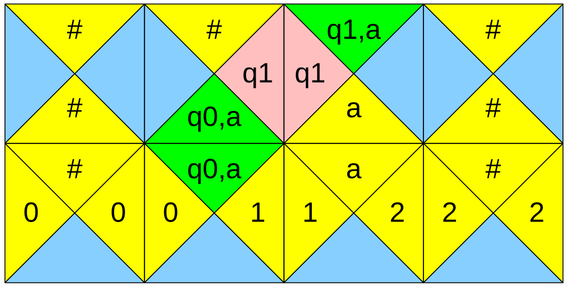


#
0
0
q0,a
0
1
a
1
2
#
2
2
#
q1
q0,a
#
q3
q0,b
#
YES
q0,#
#
q1
q1,a
#
q1
q1,b
#
q2
q1,#
#
q3
q3,a
#
q3
q3,b
#
q4
q3,#
#
NO
q2,b
#
YES
q1,#
#
NO
q4,a
#
q5
q4,b
#
YES
q4,#
a
q5
q5,a
b
q5
q5,b
#
q0
q5,#
qi,x
qi
x
qi,x
qi
x
x
x
x
NO
x
x
NO
x

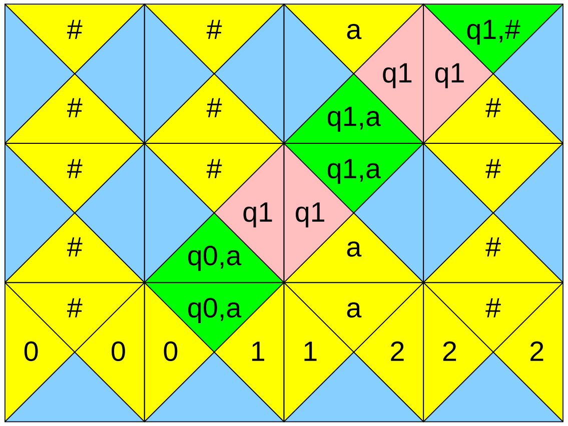
- acceptance/incomplete tiling



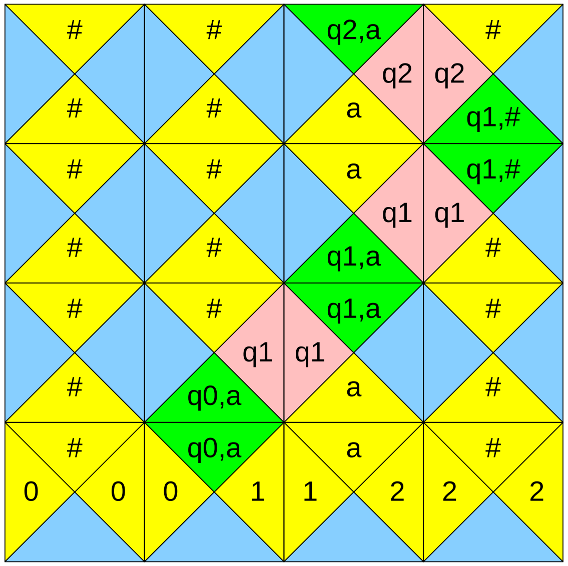
- acceptance/incomplete tiling



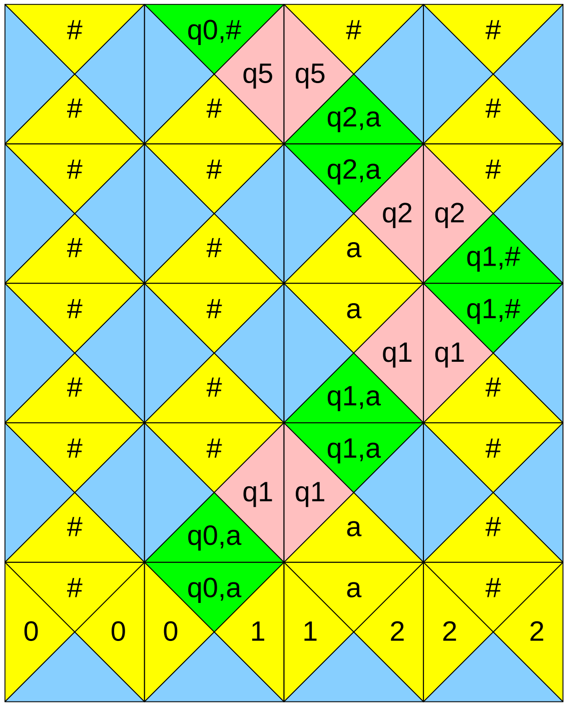
- acceptance/incomplete tiling



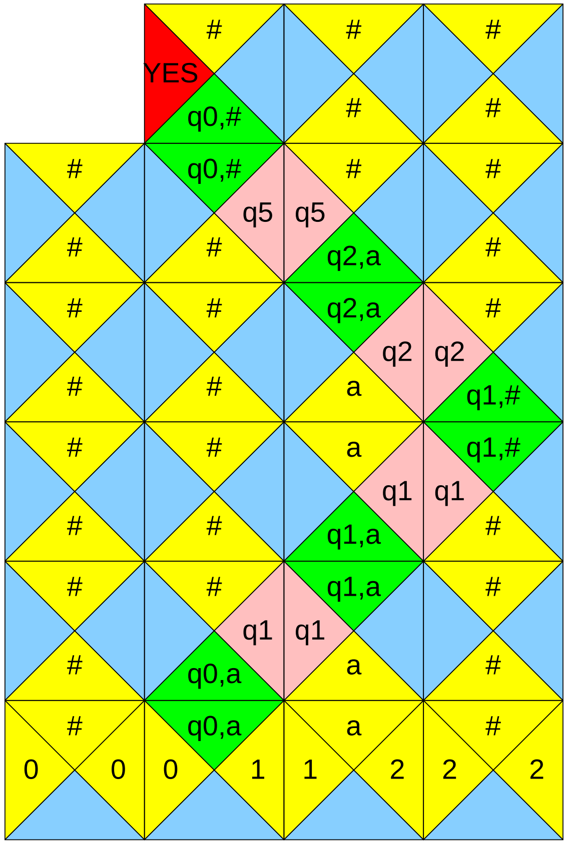
- acceptance/incomplete tiling

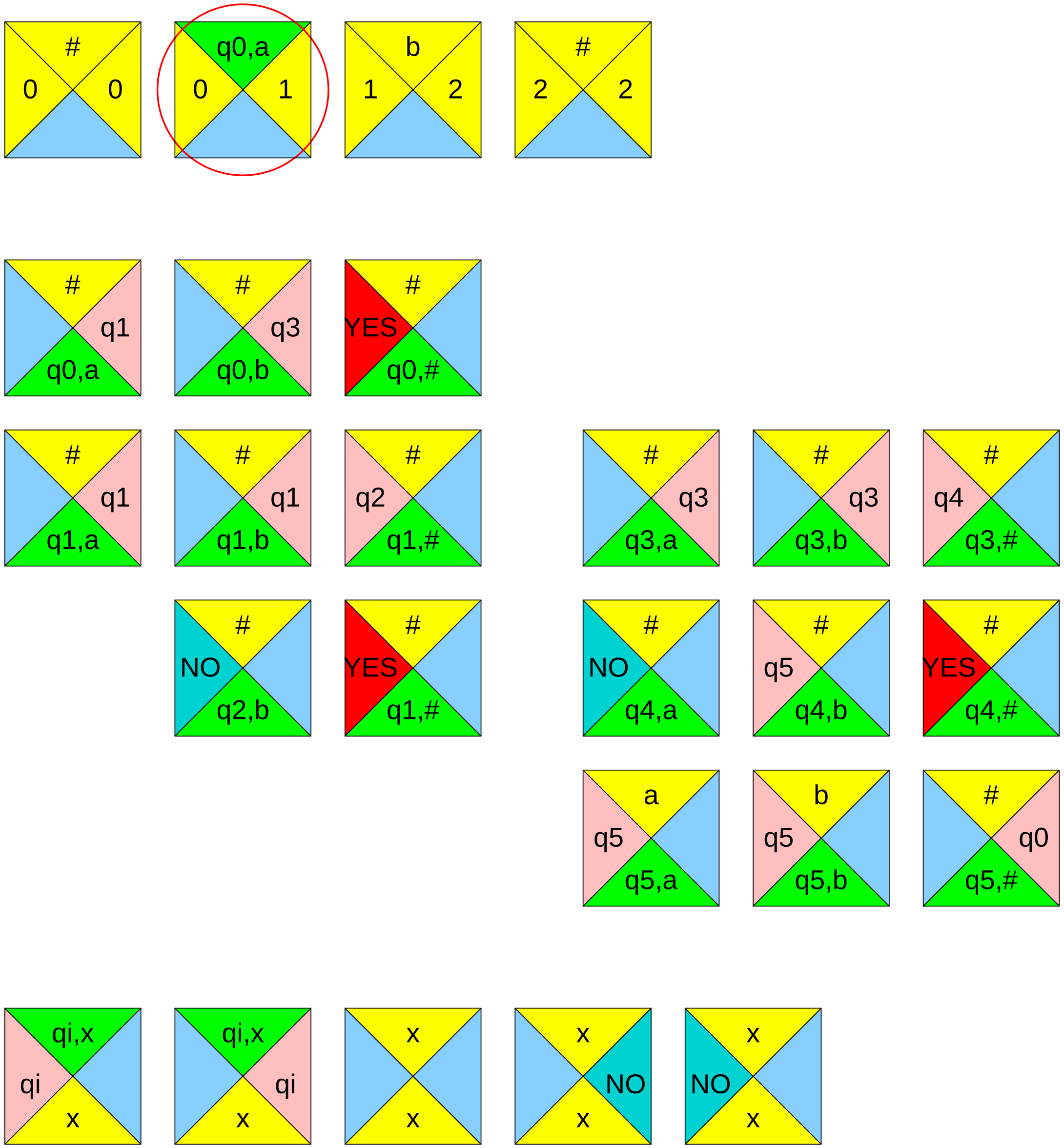


- acceptance/incomplete tiling



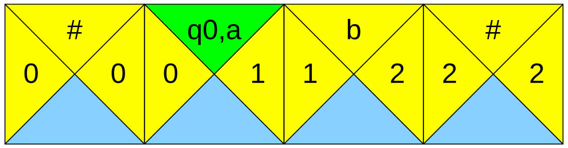
- acceptance/incomplete tiling



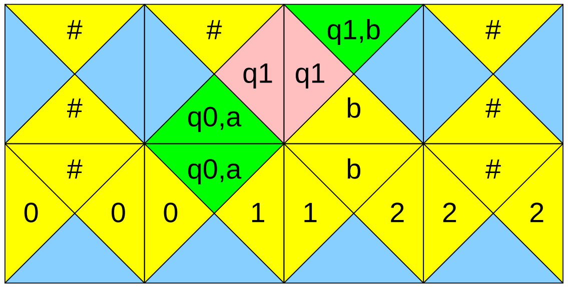


#
0
0
q0,a
0
1
b
1
2
#
2
2
#
q1
q0,a
#
q3
q0,b
#
YES
q0,#
#
q1
q1,a
#
q1
q1,b
#
q2
q1,#
#
q3
q3,a
#
q3
q3,b
#
q4
q3,#
#
NO
q2,b
#
YES
q1,#
#
NO
q4,a
#
q5
q4,b
#
YES
q4,#
a
q5
q5,a
b
q5
q5,b
#
q0
q5,#
qi,x
qi
x
qi,x
qi
x
x
x
x
NO
x
x
NO
x

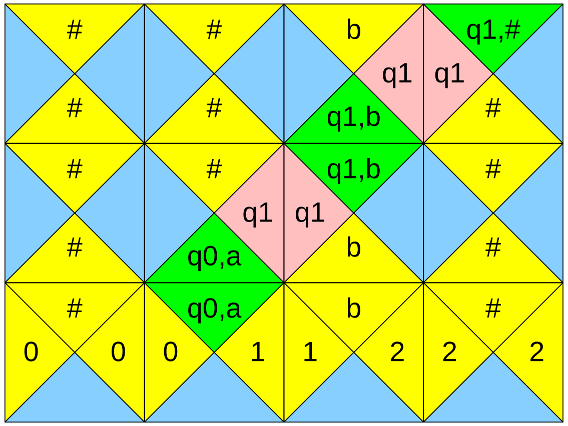
- rejection/complete tiling



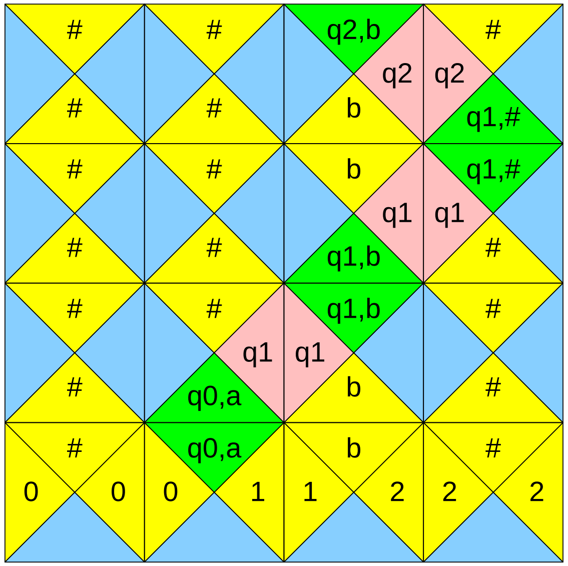
- rejection/complete tiling



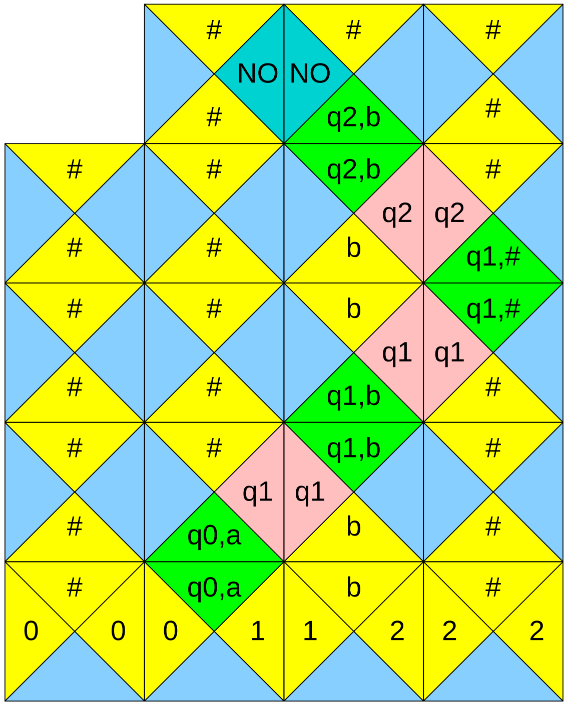
- rejection/complete tiling



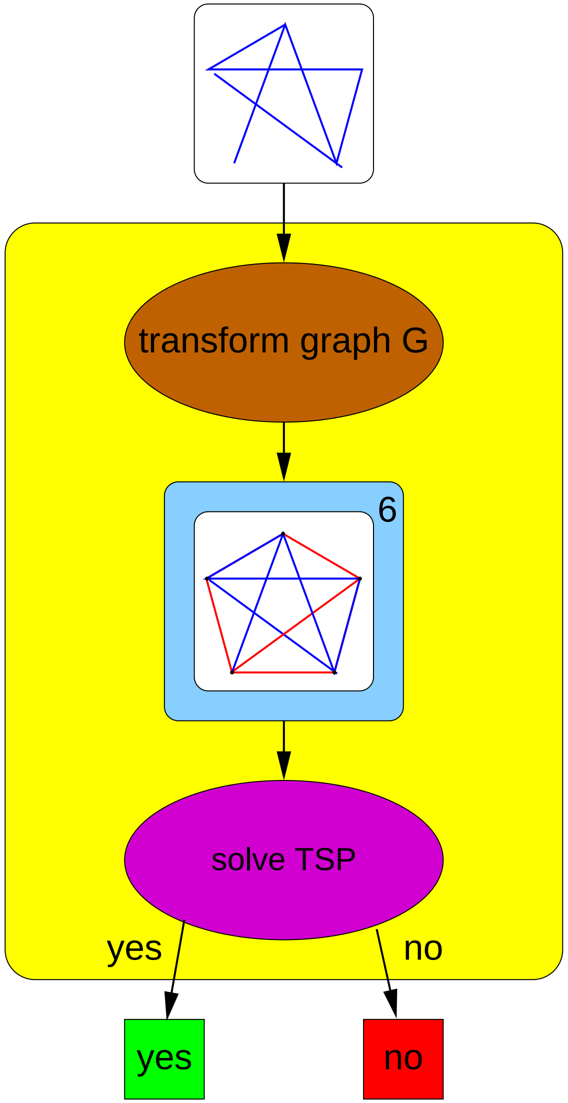
- rejection/complete tiling



- rejection/complete tiling



- reducing Hamiltonian path to travelling salesman



- reducing Hamiltonian path to travelling salesman