# SPOJ - HACK14

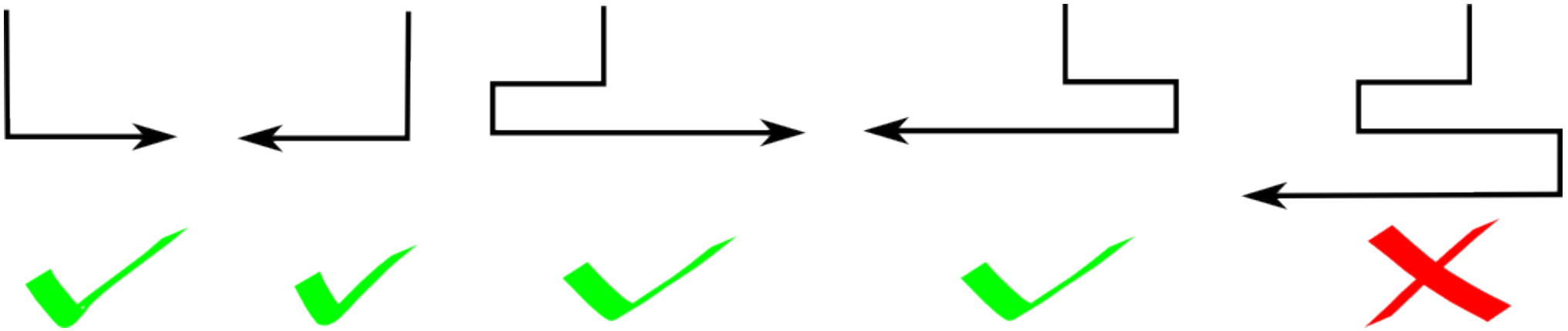
Solusi oleh M. Ayaz
Ditulis oleh Tracy and William

# Observasi 1

- Struktur optimal untuk setiap perjalanan pasti:
  - Teleport ke kota x, jalan ke kanan beberapa langkah (boleh 0 langkah), lalu ke kiri beberapa langkah, dan selesai
  - Teleport ke kota x, jalan ke kiri beberapa langkah (boleh 0 langkah), lalu ke kanan beberapa langkah, dan selesai
- Tidak mungkin perjalanannya: kiri, kanan, kiri, kanan, dst, karena tidak akan optimal

# Observasi 1 (lanj.)

- Jadi bentuknya seperti “hook”



# Observasi 2

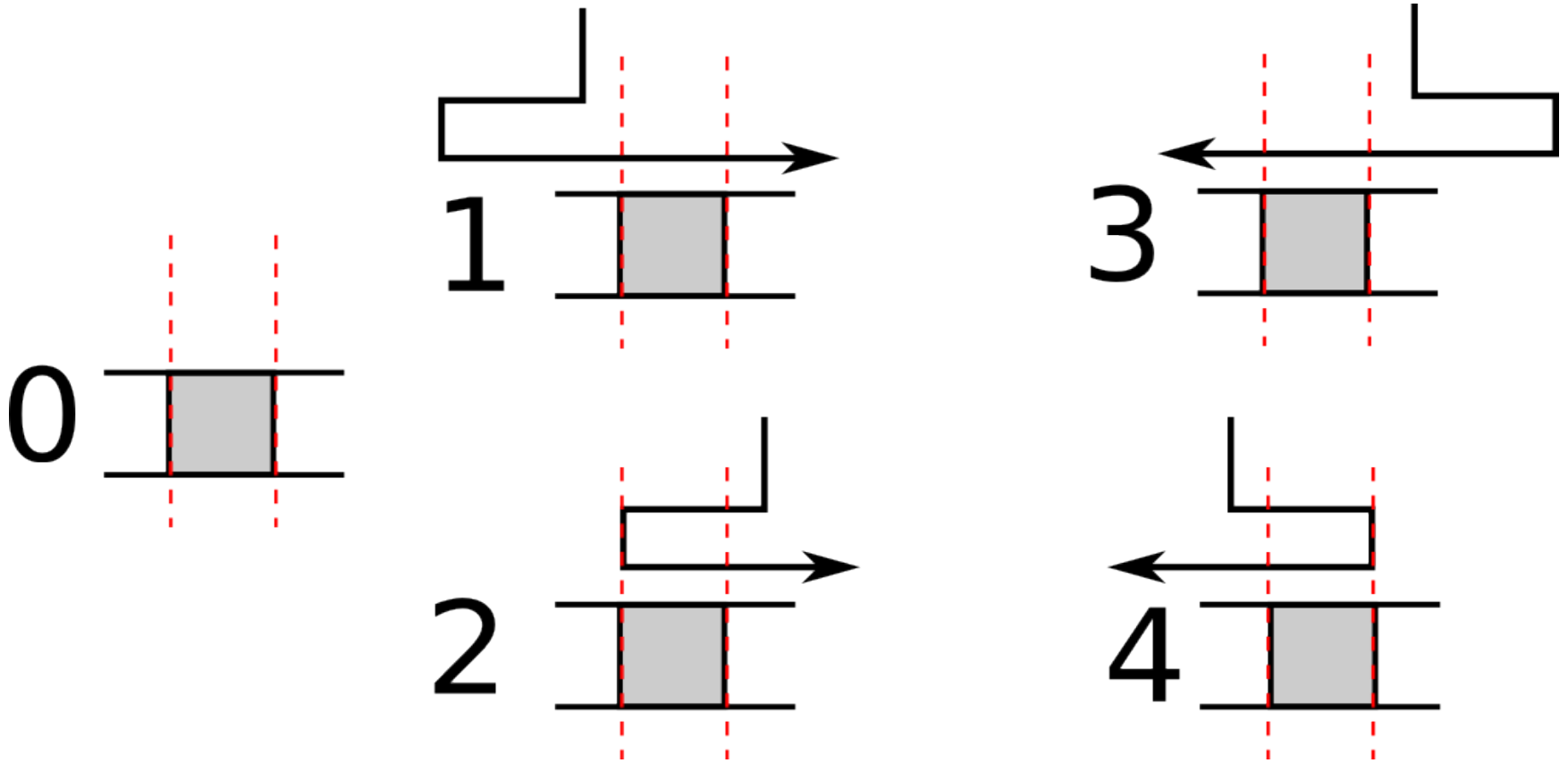
- Solusi optimal dicapai dengan membentuk maksimal K buah struktur “hook”, dan penghasilan bersihnya maksimal
- Penghasilan bersih = uang yang didapat – biaya berjalan – biaya teleport

# Ide Solusi DP Profile

- Membangun struktur “hook” dari kiri ke kanan
- Tinjau setiap kota dari kiri ke kanan
- Untuk setiap kota, tentukan peran kota tersebut supaya hasilnya maksimum

# Ide Solusi DP Profile (lanj.)

- Peran suatu kota bisa berupa:

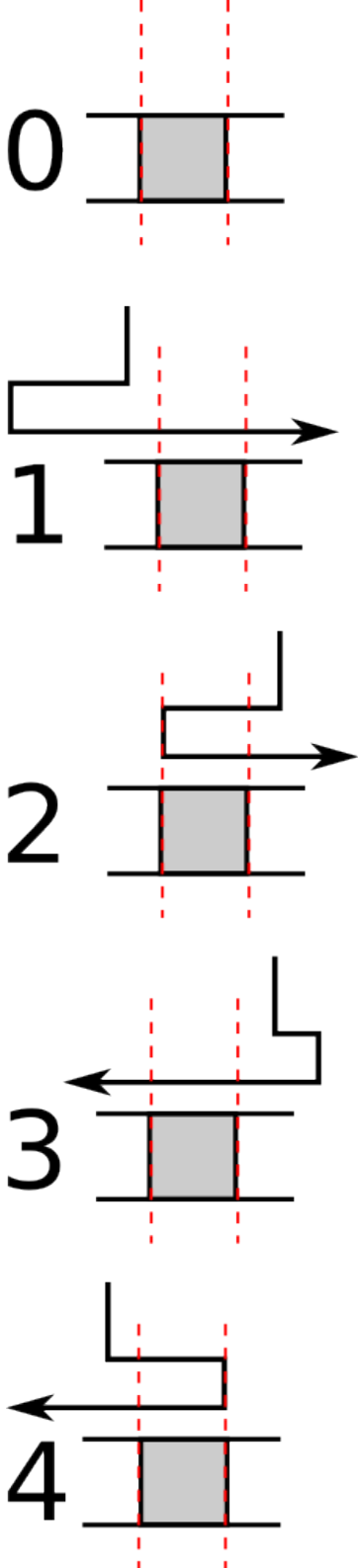


# Formulasi State

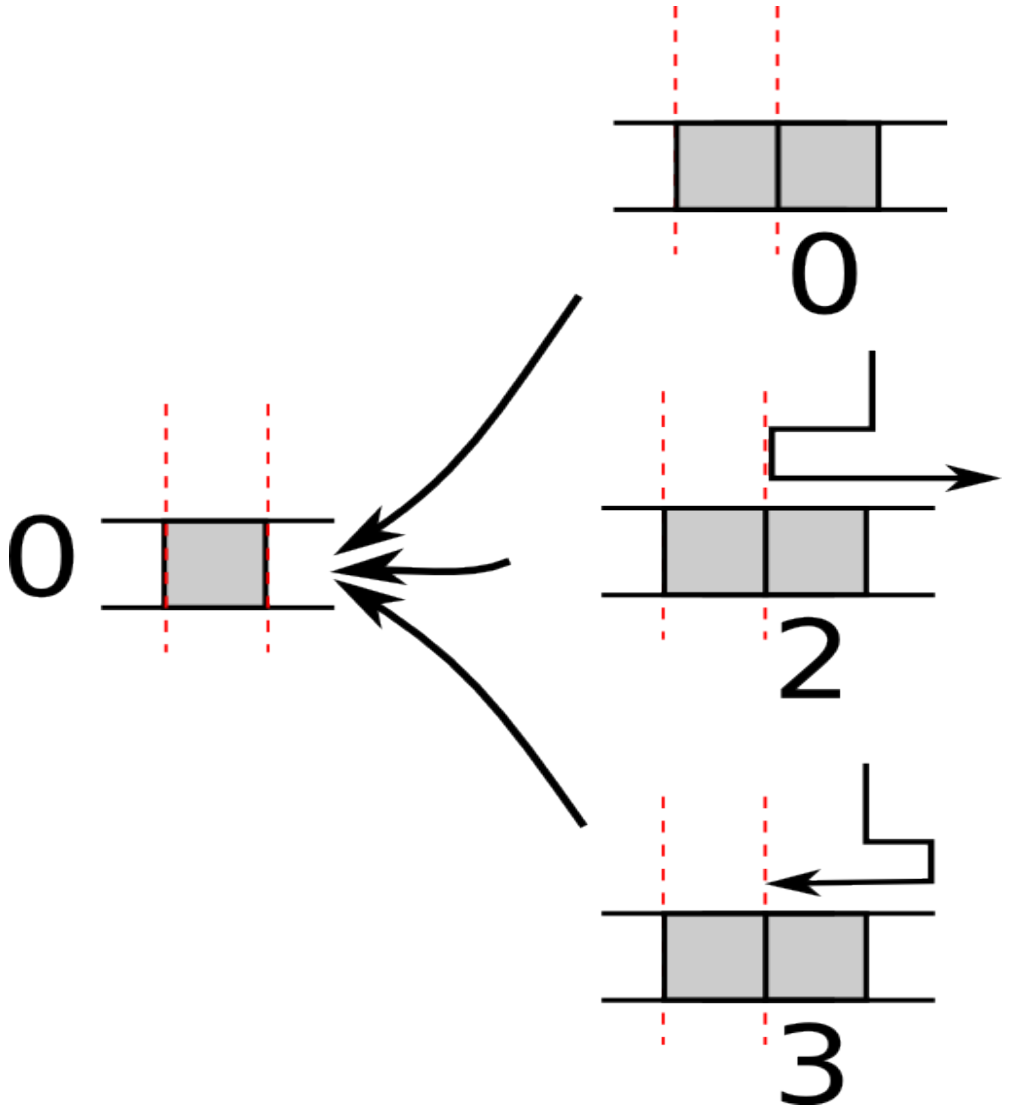
- dp(i, t, status) dengan makna solusi optimal jika ditinjau kota i..N, sisa teleport t kali, dan status sesuai dengan profile (lihat slide sebelumnya)
- Tabel DP akan diisi dari dp(n, *, *), kemudian dp(n-1, *, *), dp(n-2, *, *), …
- Solusi optimal = nilai maksimal dari dp(1, 0..K, 0/2/3)

# Base Case

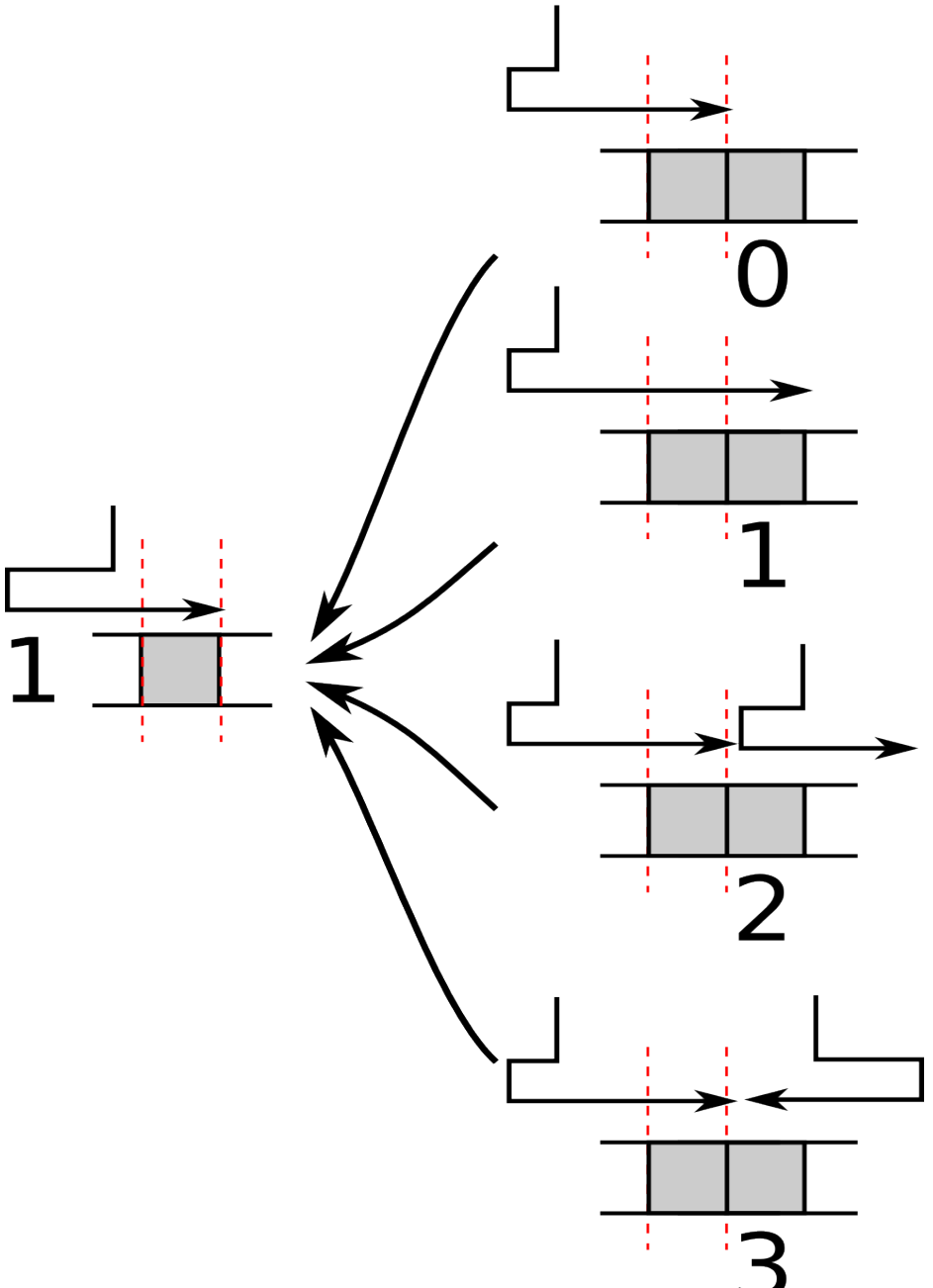
- dp(n, t, status) dengan rincian:
  - dp(n, *, 0) = 0
  - dp(n, *, 1) = G[n]
  - dp(n, 0, 2) = -INF
  - dp(n, >0, 2) = G[n] – T[n]
  - dp(n, 0, 3) = -INF
  - dp(n, >0, 3) = G[n] – T[n]
  - dp(n, *, 4) = G[n]



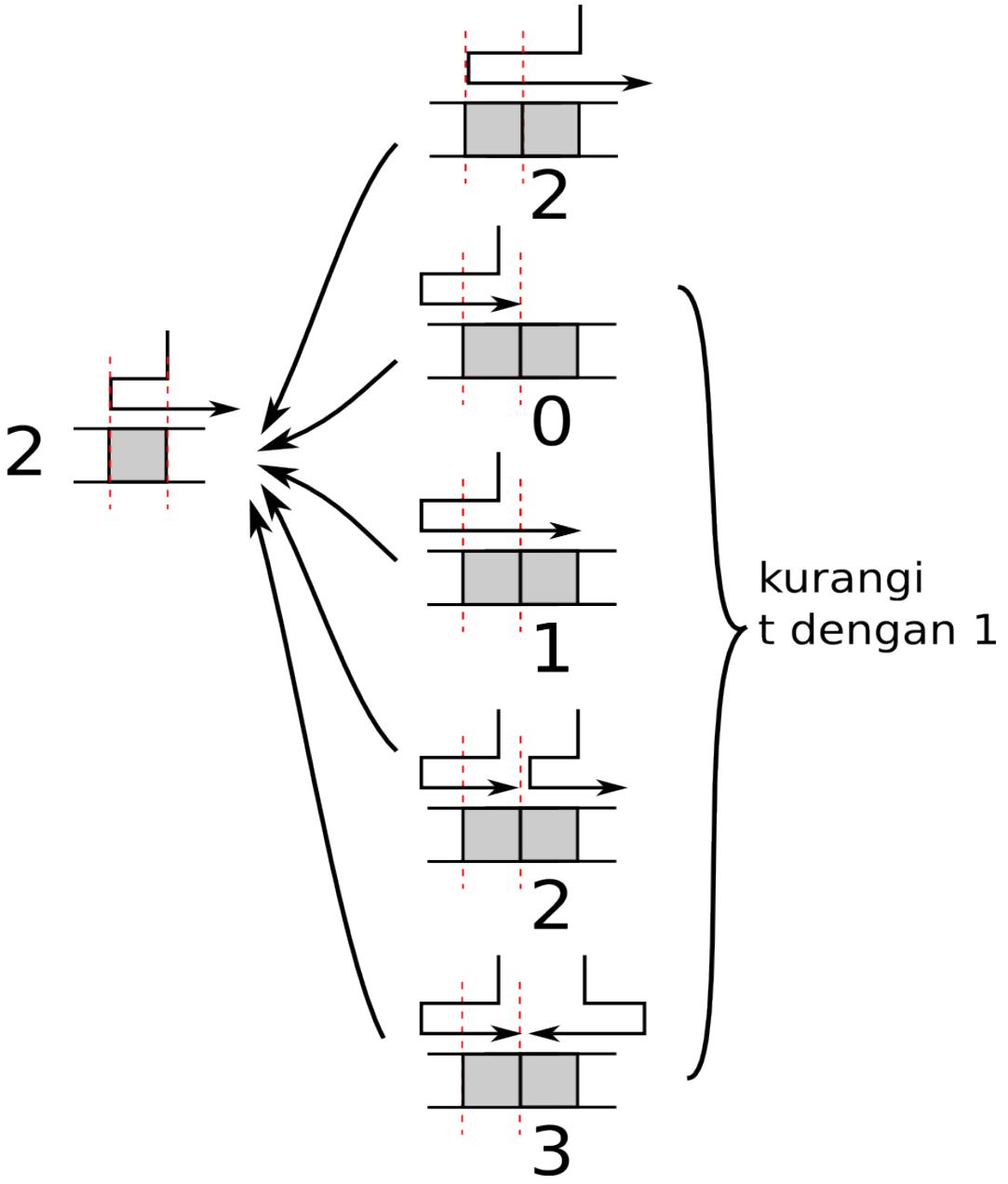
# Recurrence case



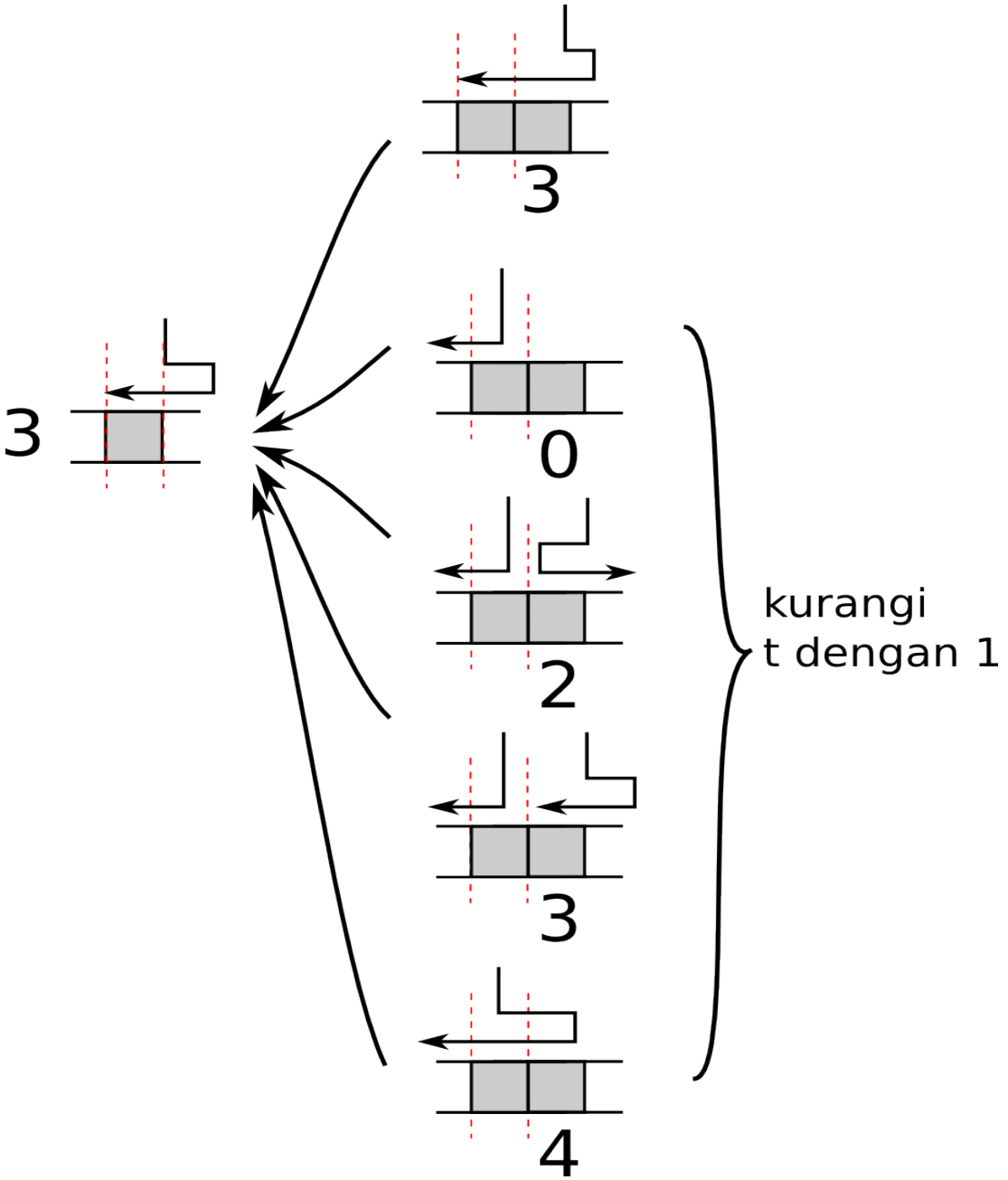
# Recurrence case



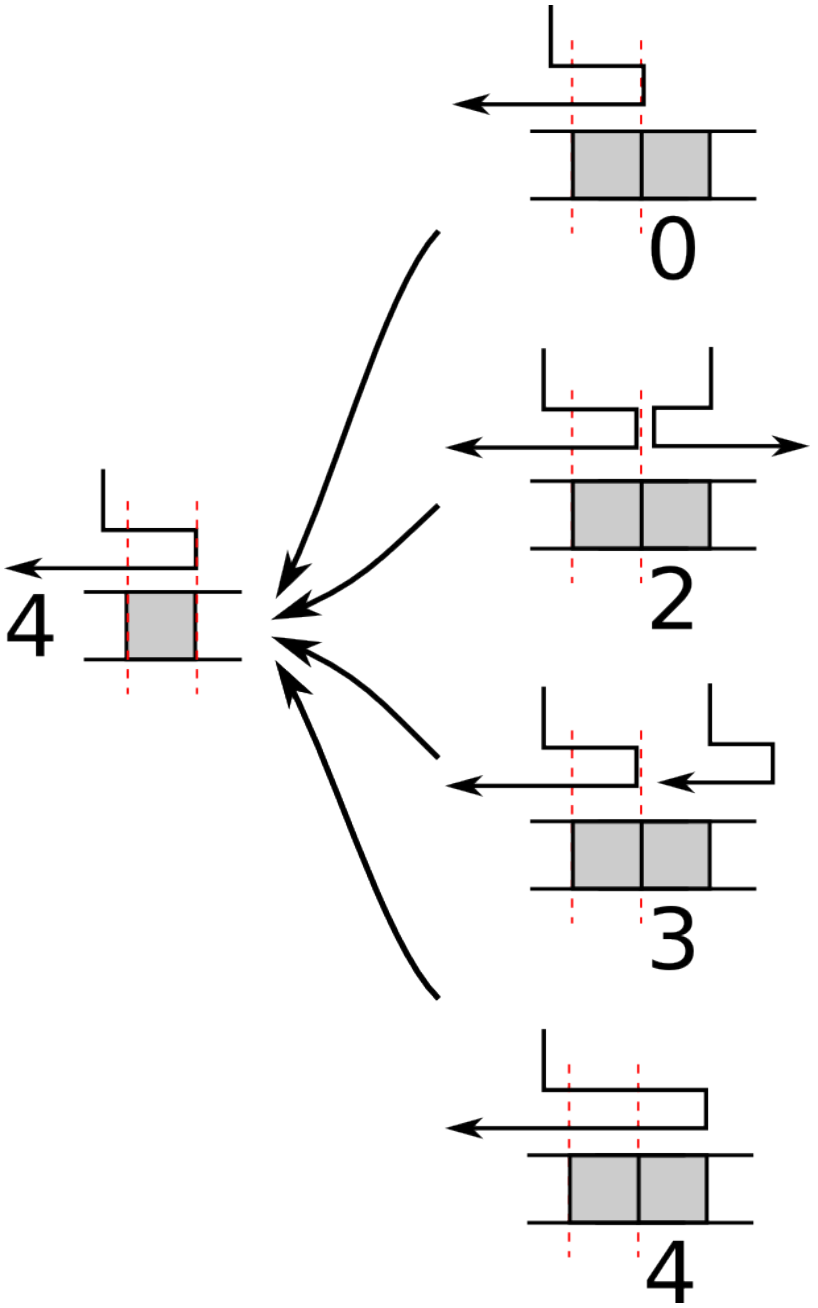
# Recurrence case



# Recurrence case



# Recurrence case



# Analisis Kompleksitas

- Banyaknya state: N*K*5
- Setiap state diisi dengan komputasi O(1)
- Total kompleksitas O(NK)

# Tambahan

- Bisa digunakan flying table
- Karena banyaknya state ~ 5 juta, sebaiknya gunakan bottom up