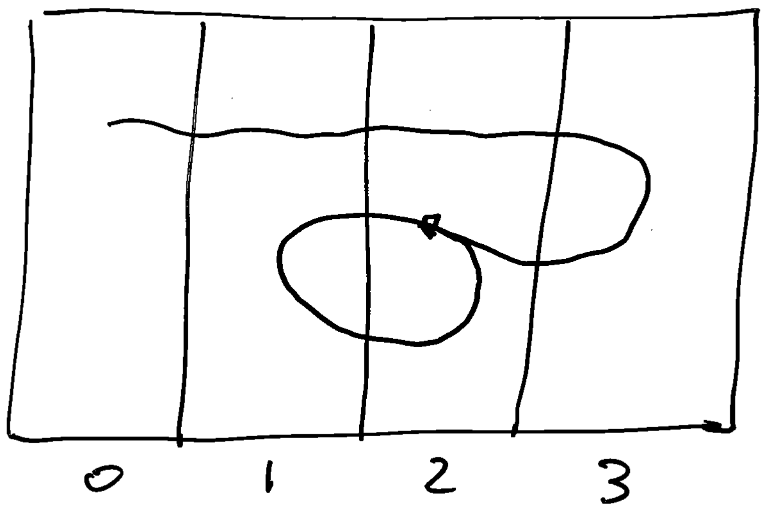


S = set of states (finite) $\sigma \in S^\omega$.

$\mathrm{infi}(\sigma) = \{ s \in S \mid s$ appears ∞ often in σ

-or-

$\exists$ infinitely many $k \in \mathbb{N}$ s.t. $\sigma_k = s$

-or-

$\forall i \in \mathbb{N}.\ \exists k > i.\ \sigma_k = s \}$.

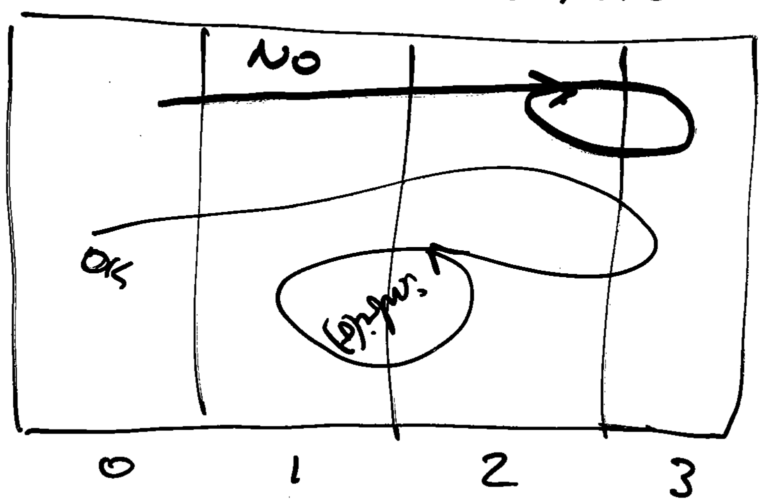
HW

$\sigma \overset{\in}{\models} \Diamond\Box\, \mathrm{infi}(\sigma)$.

satisfies, or "has the property".

$\sigma \models \varphi$ iff $\sigma \in L(\varphi)$

PARITY CONDITION



~~σ is ok~~

Parity condition: $\gamma : S \mapsto \mathbb{N}$.

σ is accepted by γ iff

$$\max \left(\gamma(s) \mid s \in \mathrm{infi}(\sigma) \right) \text{ is even.}$$

"the ~~k~~ right edge of $\mathrm{infi}(\sigma)$ is in an even slice".

Given γ, we have $L(\gamma) = \{ \sigma \mid \sigma \text{ accepted by } \gamma \}$.

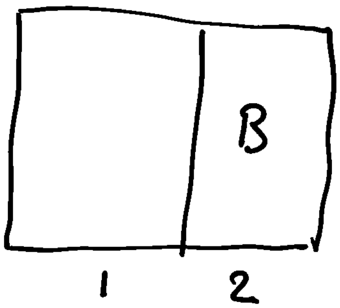
$\neg L = S^\omega \setminus L = L(\gamma + 1)$

where $(\gamma + 1)(s) = \gamma(s) + 1$.

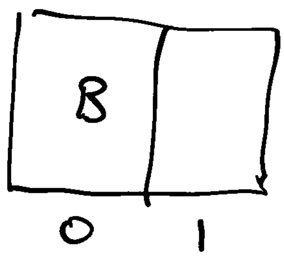
Why are parity conditions important?

- Easy to complement.
- Any ω-regular property can be expressed via a deterministic automaton with parity condition.
- Linear temporal logic $\subseteq$ ω-regular properties.
- μ-calculus $\rightarrow$ parity automata.

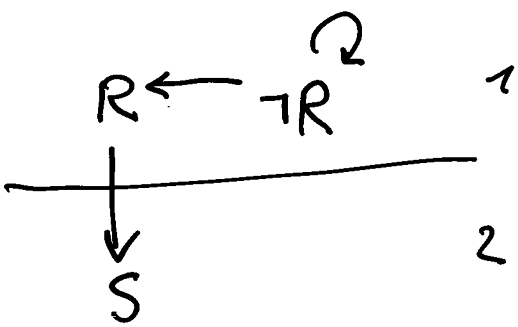
Büchi or parity: $\Box\Diamond B$



Co Büchi $\Diamond\Box B$

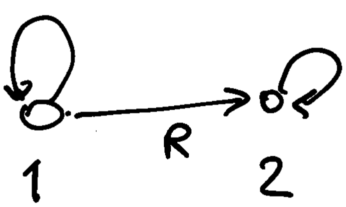


Reach

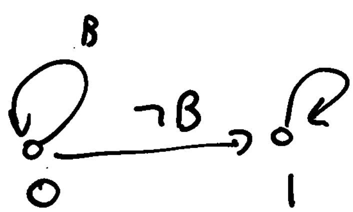


not reached

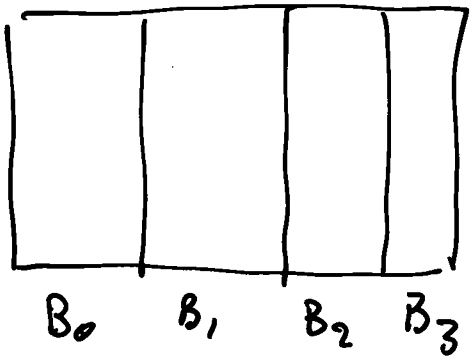
reached



Safety $\Box B$



Sol. 1 : μ-calculus.



$$B_i = \{s \mid \gamma(s) = i\}$$

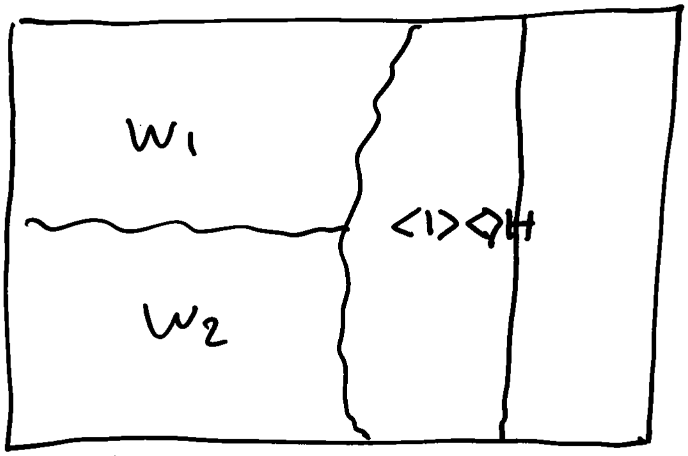
$$\langle 1 \rangle \; Parity(\gamma) = \mu X_3 . \nu X_2 . \mu X_1 . \nu X_0 . \begin{pmatrix} B_0 \wedge Pre_1(X_0) \\ \vee \\ B_1 \wedge Pre_1(X_1) \\ \vee \\ \cdots \\ \vee \\ B_3 \wedge Pre_1(X_3) \end{pmatrix}$$

$$d = \max\{\gamma(s) \mid s \in S\}$$

Emerson
Jutla 91

$$O(|S|^d).$$

Divide and Conquer – (Recursive)



$\langle 2 \rangle \Box \neg H$ $\qquad$ H = top slice (even)

- Compute $\langle 1 \rangle \Diamond H$ "attractor of H".
  if $S = \langle 1 \rangle \Diamond H$, then $\langle 1 \rangle \gamma = S$.

What is left over is $\underbrace{\langle 2 \rangle \Box \neg H}$.
This is a subgame:

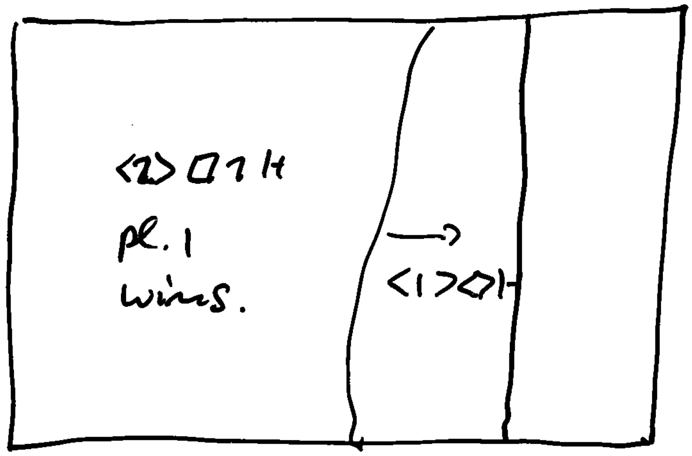
Consider:
- all moves of pl. 1
- all moves of pl. 2 which do not leave $\langle 2 \rangle \Box \neg H$.

Solve it. Compute W_1 = pl. 1 wins in subgame
W_2 = pl. 2 wins in subg.

- If $W_2 = \emptyset$, then $\langle 1 \rangle \gamma = S$.

Because: - If pl. 2 keeps the game in $\langle 2 \rangle \Box \neg H$, pl. ~~2~~ 1 can win there.
- If pl. 2 allows ∞ many visits to H, pl. 1 also wins there.

2 — Divide and Conquer-(Recursive)



$W_1 = \langle 2\rangle \Box \neg H$. H = top slice (even)

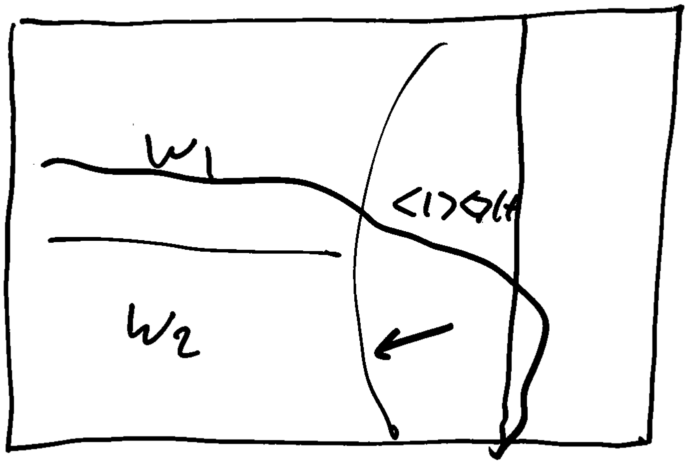
Pl. 1 plays as follows:

- In $\langle 2\rangle \Box \neg H$, pl. 1 plays the strategy that wins in the subgame.
- In $\langle 1\rangle \Diamond H$, pl. 1 plays to reach H.

<u>Two cases:</u>

- Pl. 2 keeps game forever in $\langle 2\rangle \Box \neg H$.
  $\longrightarrow$ pl. 1 wins.
- Pl. 2 often lets the game out of $\langle 2\rangle \Box \neg H$. then, $\Box\Diamond H$, $\leadsto$ 1 wins.
- If the game forever is in the subgame, pl. 1 wins in the subgame.

2 — Divide and Conquer - (Recursive) OV5



H = top slice (even)

- If $W_2 \neq \emptyset$, compute $\langle 2 \rangle \Diamond W_2$.

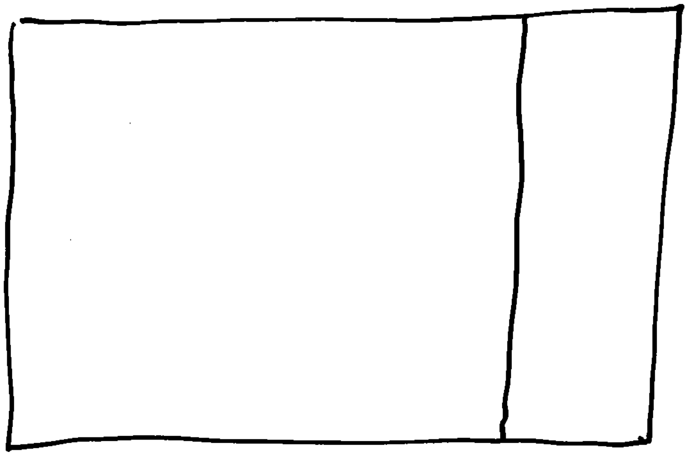
  Then, $\langle 2 \rangle \Diamond W_2$ is a "dominion": pl 2 can win in there.

  we know $\langle 2 \rangle \Diamond W_2 \subseteq \langle 2 \rangle \neg \gamma$.

- Shave of $\langle 2 \rangle \Diamond W_2$, and ~~go a~~ call the algo again on

  $\langle 1 \rangle \Box \neg W_2$.

2 — Divide and Conquer (Recursive)



d slices

$|S| =$ m edges
n states.

H = top slice (even)

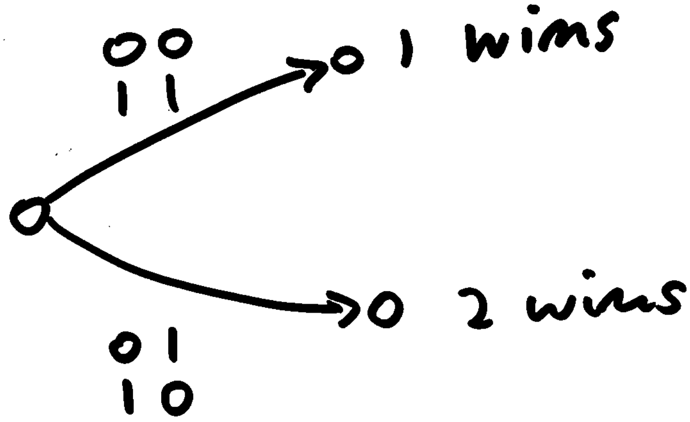
Summary: solve (S) $\qquad T(n)$

- Compute $\langle 2\rangle \Box \neg H$ $\qquad O(m)$
  empty $\to$ ~~stop~~ return "S"
- Compute W_1, W_2 in $\langle 2\rangle \Box \neg H$ $\qquad T(n-1)$
  $W_2 = \emptyset \to$ return "S".
- Compute $L = \langle 2\rangle \Diamond W_2$. $\qquad O(m)$
- Return solve $(S \setminus L)$. $\qquad T(n-1)$

$$T(n) \leq O(m) + 2T(n-1)$$

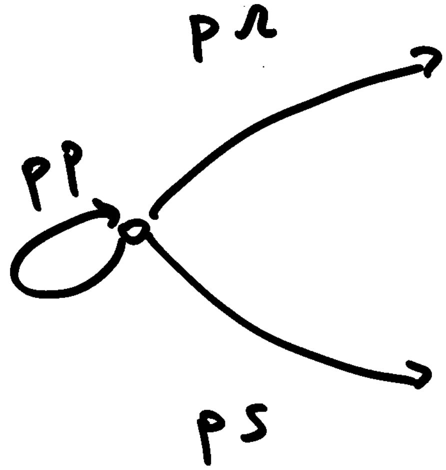
$$T(n) = O(m \cdot 2^{d})$$

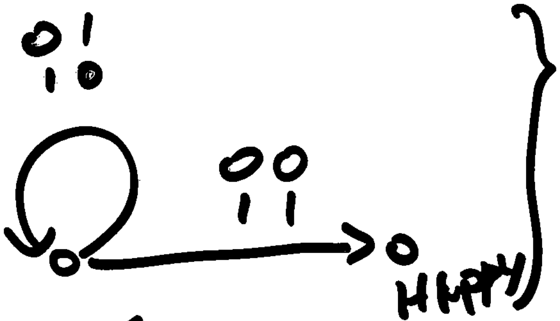
More careful: $O\left(\frac{n}{d}\right)^d$.



0,1
0,1

paper
scissors
~~stone~~
rock





eventual
penny
matching

DetStrat

0 1 0 0 1 1 0 1 0 1
1 0 1 1 0 0 1 0 1 0 ...

$\forall \pi_1 \exists \pi_2 .$ pl. 2 wins $(\Box \neg$ happy$)$

$\neg \exists \pi_1 \forall \pi_2 . \Diamond$ happy