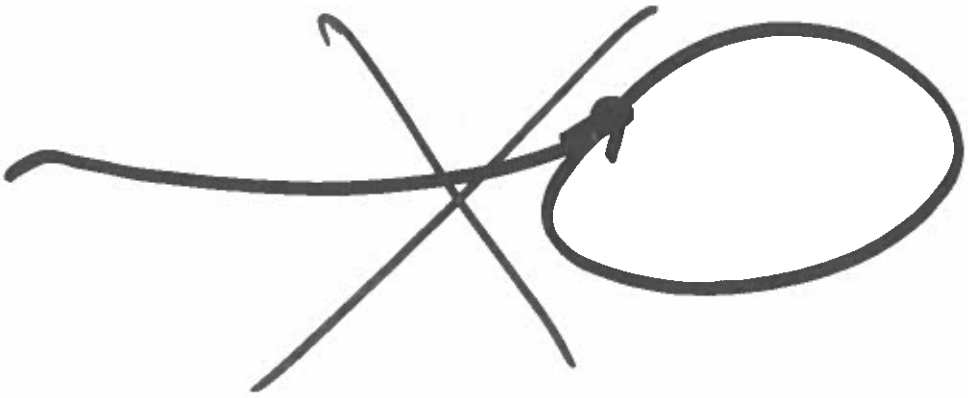
# FROM COMPLEMENTATION TO CERTIFICATION

Orna Kupferman
Hebrew University

Moshe Y. Vardi
Rice University

Today's lesson:

How not to run in circles!



# Why complementation?

## Language-containment perspective for verification

Implementation – set of computations

specification – set of computations

verification: $Imp \subseteq spec$

$\Updownarrow$

$Imp \wedge \overline{spec} = \emptyset$

# Automata on Finite Words

$A = (\Sigma, S, S_0, \rho, F)$

$\Sigma$: finite alphabet

$S$: finite state set

$S_0 \subseteq S$: initial state set

$\rho : S \times \Sigma \to 2^S$ : transition function

$F \subseteq S$: accepting state set

Input: $a_0 \dots a_{n-1}$

Run: $r_0 \dots \dots r_n$

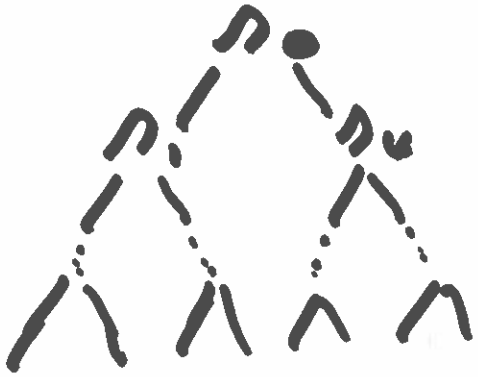
- $r_0 \in S_0$
- $r_{i+1} \in \rho(r_i, a_i)$

Acceptable: $r_n \in F$

# COMPLEMENTATION I

**Input:**


**Run tree:**



**Run DAG**

subset construction:

$$A^c = (\Sigma, 2^S, \{s_0\}, \delta^c, F^c)$$

$$F^c = \{T : T \cap F = \emptyset\}$$

$$\delta^c(T, a) = \bigcup_{t \in T} \delta(t, a)$$

# Automata on Infinite Words

Looping automaton:

$A = (\Sigma, S, S_0, \rho)$

Input: $a_0, a_1, \ldots$

Run: $r_0, r_1, \ldots$

- $r_0 \in S_0$
- $r_{i+1} \in \rho(r_i, a_i)$

Acceptance: $\exists$ infinite run

Motivation:

- Infinite computations
- Safety properties

# Complementation II

**Input** **Run Tree** **Run DAG**

$a_0$
$a_1$
$\vdots$

Subset Construction:

$$A^c = (\Sigma, 2^S, \{s_0\}, \rho^c, \emptyset)$$

$$\rho^c(T, a) = \bigcup_{t \in T} \rho(t, a)$$

automaton on finite words

# Büchi Automata

$A = (\Sigma, S, S_0, \rho, F)$

Input: $a_0, a_1, a_2 \ldots$

Run : $s_0, s_1, s_2 \ldots$

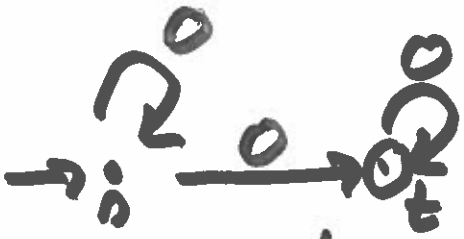
Acceptance: run visits $F$ infinitely often

Motivation:

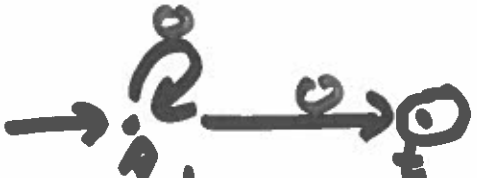
- $\omega$-regular properties
- LTL $\rightsquigarrow$ BA

# Complementation III

subset construction fails!




accepts




not accepts

subset construction:

# History

- Büchi, 1962: $2^{2^n}$
- Sistla+Vardi+Wolper, 1985: $2^{n^2}$
- Safra, 1988: $n^n$
- Klarlund, 1991: $n^n$
- Kupferman+Vardi, 1997: $n^n$
- Merz, 2000: HOL implementation
- GKSV, 2003: optimization

Note: Big O ignored.

# Generalized Büchi Automata

$A = (\Sigma, S, S_0, \rho, (F_1 \dots F_k))$

$F_i \subseteq S$: accepting states

Input: $a_0 a_1 a_2 \dots$

Run: $s_0 s_1 s_2 \dots$

Acceptance: run visits each $F_i$ infinitely often

Motivation:

- More expressive fairness
- Easier translation from LTL

# Complementing GBA

Known: $GBA \to BA$

$A = (\Sigma, S, s_0, \rho, (F_1 \dots F_k)) \mapsto A' = (\Sigma, S', \dots)$

$|S| = n \qquad |S'| = kn$

Complementing GBA: $(kn)^{kn}$

Our improvement: $(kn)^{n}$

Exponential improvement

# From "No" to "Yes"

Traditional Model checking:

$M$: system

$\varphi$: specification

1. $M \not\models \varphi$: provide counterexample.
2. $M \models \varphi$: say "yes".

Recent emphasis: say more than "yes"

- Coverage: Enough properties?
- Vacuity: Genuine "yes"?
- Certification: prove it!

# Certification

cert$(M, \varphi)$ : proof that $M \models \varphi$.

Motivation:

- Did you check your checker!
- Independent or outsourced components
- Proof-carrying code

Prior Work: deductive proofs

- Namjoshi, 2001
- Peled, Pnueli, Zuck, 2001

Our emphasis: compact certificates

# Transition systems

$M = (W, W_0, R)$

$W$: state set

$W_0 \subseteq R$: initial state set

$R \subseteq W^2$: transition relation

Invariance checking:

$G \subseteq W$

show that $Reach(W_0, R) \subseteq G$.

Fact: Model checking of safety properties can be reduced to invariance checking.

# Invariance Certification

Invariance checking:

Reach := $W_0$
while change do
  Reach := Reach $\cup$ Post(R, Reach)
Return  Reach $- G = \emptyset$

Certificate: Reach

Certificate checking: $U \subseteq W$

- $W_0 \subseteq U$
- $U \cup Post(R, U) = U$
- $U - G = \emptyset$

# Fair Termination

$M = (W, W_0, R)$

$F \subseteq W$

Fair termination: No infinite path in $M$ from $W_0$ visits $F$ infinitely often — essence of linear-time model checking

## EL Algorithm (EC'80)

```
Q := W
while change do
    Q := Q ∩ until_R(Q, Q∩F)
od
return W0 ∩ Q = ∅
```

Certificate? Q?

# Complementation IV

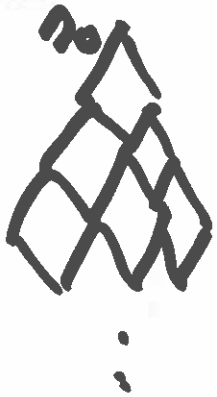
$A = (\Sigma, S, s_0, \rho, F)$ : BA

Input

$a_0$
$a_1$
⋮

Run Tree

Run DAG



Acceptance: No infinite path visits F infinitely often

EL: $Q := $ all nodes
while change do
$Q := Q \cap \mathrm{until}(Q, Q \cap F)$
od
Return $S_0 \cap Q = \emptyset$

Convergence: transfinite!

# Termination

$M = (W, W_0, R)$

Termination: No infinite path in $M$ from $W_0$.

old Algorithm*: inductive dfn of well-foundedness

```
Q := W
while change do
    Q := Q ∩ Pre(R, Q)
od
Return W0 ∩ Q = ∅
```

* Used by symbolic model checkers to eliminate finite paths.

# Fair Termination Revisited

$M = (W, W_0, R)$

$F \subseteq W$

Fair termination: No infinite path in $M$ from $W_0$ visits $F$ infinitely often.

**OWCTY** (FFKVY'01)

```
Q := F
while change do
    while change do
        Q := Q ∩ Pre*(R, Q)
    od
    Q := Q ∩ until_R(Q, Q ∩ F)
od
return W_0 ∩ Q = ∅
```

Debate: EL vs. OWCTY

# Complementation ✓

$A = (\Sigma, S, S_0, \rho, F)$ : BA $|S| = n$

Input

$a_0$
$a_1$
⋮

Run Tree

Run DAG


Acceptance: no infinite path visits infinitely often.

```
WCTY: Q := all nodes
      while change do
        while change do
          Q := Q ∩ Pre(A, Q)
        od
        Q := Q ∩ Until_A(Q, Q ∩ F)
      od
      return Σ0 ∩ Q = ∅
```

Convergence: $n+1$ external iterations!

# Ranks

## Run DAG


## OWCTY

```
Q := all nodes
while change do
  while change do
①   Q := Q ∩ Pre(Q, Q)
  od
②Q := Q ∩ Until(Q, Q∩F)
od
```

Acceptance: $Q = \emptyset$

**Ranks:**

- rank(x) = 2i: x deleted in ① at iteration i.
- rank(x) = 2i+1: x deleted in ② at iteration i.

**Maximum rank: 2n**

Related: Klarlund's progress measures 21

# Ranks and Complementation

$A = (\Sigma, S, S_0, \rho, F)$, $|S| = n$

Rank assignment: $S \to \{0, \ldots, 2n\}$

$\alpha = \{0, \ldots, 2n\}^S$: all rank assignments

$A^c = (\Sigma, 2^S \times \alpha, \{S_0\} \times \alpha, \rho^c, F^c)$

$\rho^c$:

- ranks cannot increase
- cannot get stuck forever in an even rank
- F-states cannot get odd ranks

Consequence: cannot visit F infinitely often.

Complexity: $n^{O(n)}$ states

# Ranks vs. Certification

## Basic idea

1. Augment OWCTY to compute ranks as it deletes nodes
2. Final rank assignment is the certificate (see Namjoshi).
3. Use algebraic decision diagrams (ADDs) as data structure.
4. Certificate can be checked using a fixed number of ADD operations.

Note: pseudocode in proceedings buggy; see Web. (Zuck)

# In Conclusion

Key ideas:

1. OWCTY yields bounded rather than transfinite convergence.
2. Bounded ranks are crucial to complementation.
3. Rank assignments can be used as certificates.

Future: other types of $\omega$-automata – Rabin, Streett