

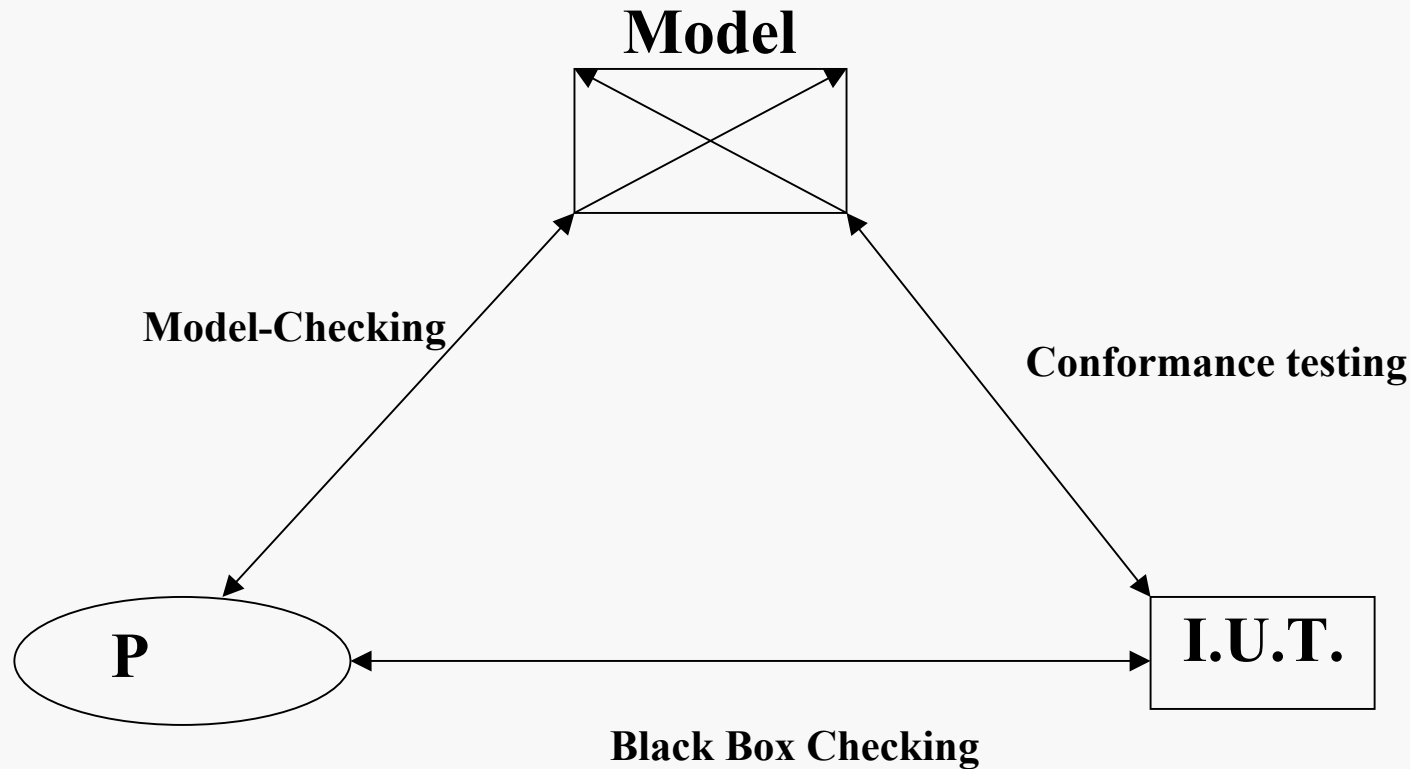
Approximate verification (part 2)

Michel de Rougemont, LRI , *University Paris II*

Joint work with E. Fischer, *Technion*, F. Magniez, *LRI*



Model Checking and Test



Example: finite automata, $P: 0^*1^*$

When exact verification is too hard, approximate.

Approximate Verification

1. Probabilistic verification
Interactive Proofs, PCP, Property testing
2. Approximate equality (constant time)

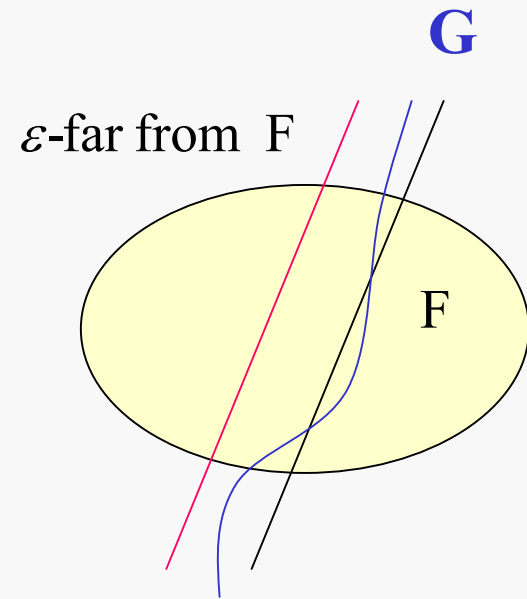
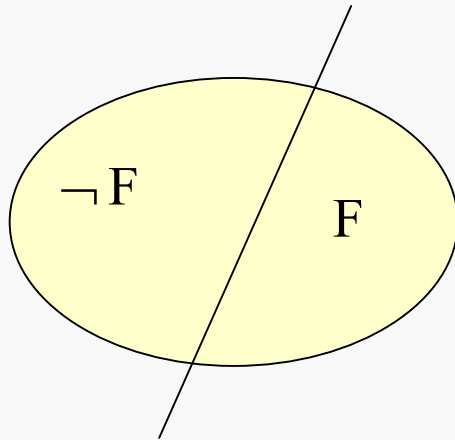
End of part 1

3. Test if a word w is in a regular language (constant time)
Test if a tree T is in a regular Tree language (constant time)
4. Equivalence Testing: Compare two languages
Finite automata (regular expressions): Polynomial time approximation (exact version is PSPACE complete)
Buchi Automata: Polynomial time approximation
Extensions to CF, Regular Expression with squaring and negation (Meyer, Stockmeyer 1972)

Approximate Satisfiability and Equivalence

1. Satisfiability : $\text{Tree} \models F$
2. Approximate satisfiability $\text{Tree} \models_{\varepsilon} F$
3. Approximate equivalence $F \equiv_{\varepsilon} G$

Image on a class K of trees



Testers on a class K

Let F be a property on a class K of structures U

An ε -tester for F is a probabilistic algorithm A such that:

- If $U \models F$, A accepts
- If U is ε far from F , A rejects with high probability
- $\text{Time}(A)$ independent of n .

(Goldreich, Golwasser, Ron 1996 , Rubinfeld, Sudan 1994)

Tester usually implies a linear time corrector.

Tester for equality of strings (part 1)

Edit distance with moves. NP-complete problem, but $O(1)$ -approximable.

Uniform statistics ($k=\frac{1}{\varepsilon}$): $\mathbf{W}=\mathbf{001010101110}$ $u.stat(W)=\begin{pmatrix} 1 \\ 4 \\ 4 \\ 2 \end{pmatrix} \cdot \frac{1}{11}$

Theorem 1. $|u.stat(w)-u.stat(w')|$ approximates $dist(w,w')/n$.

Sample N subwords of length k , compute $Y(w)$ and $Y(w')$:

$$Y(w)=\frac{1}{N} \sum_{i=1\dots N} X_i \quad Y(w')=\frac{1}{N} \sum_{i=1\dots N} X_i \quad X_i = \begin{pmatrix} 0 \\ 1 \\ 0 \\ \ddots \\ 0 \end{pmatrix}$$

Theorem 2. $Y(w)$ approximates $u.stat(w)$.

Corollary. $|Y(w)-Y(w')|$ approximates $dist(w,w')/n$.

Tester: If $|Y(w)-Y(w')| < \varepsilon$. accept, else reject.

Tester for regular words

Tester. Input : W, A, ε

For $i=1, \dots, \log(m/\varepsilon)$

Choose $N_i = \Theta(2^{-i} \cdot m^3 / \varepsilon)$ random
subwords w_j^i of size 2^{i+1}

For every admissible path Z :

If all w_j^i of W are Z feasible, ACCEPT.
else REJECT.

Theorem: $\text{Tester}(W, A, \varepsilon)$ is an ε -tester for $L(A)$.

Proof schema of the Tester

Theorem: Regular words are testable.

Robustness lemma: If W is ε -far from L , then for every admissible path Z , there exists $i \leq \log\left(\frac{5 \cdot m^2}{\varepsilon}\right)$ such that the number

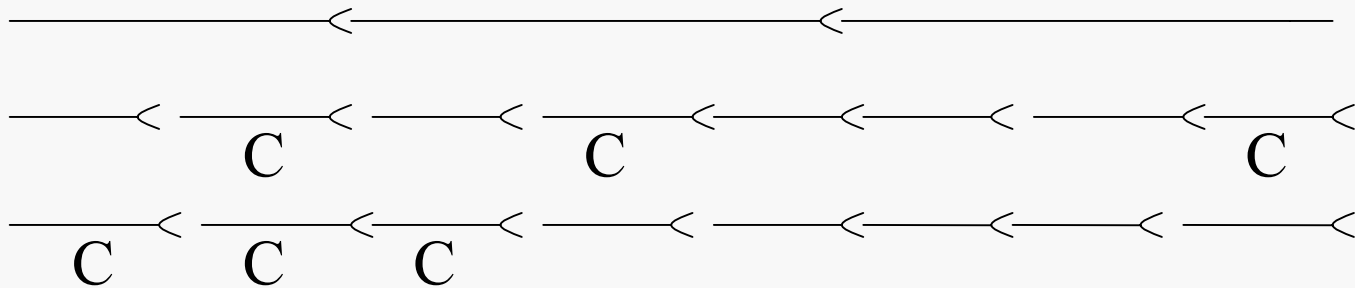
of Z -infeasible subwords of length 2^{i+1} is at least $\frac{2^{i+1}}{m^2} \cdot \varepsilon \cdot n$.

Splitting lemma: if W is far from L there are many disjoint infeasible subwords.

Amplifying lemma: If there are many infeasible words, there are many short ones.

Merging trees

Merging lemma: Let Z be an admissible path, and let F be a Z -feasible cut of size h' . Then $Dist(F,L) \leq m^2 h'$



Take each word $w_i \in F$ and split it along its connected components, removing single letters. Rearrange all the words of the same component in its Z -order.

Add gluing words to obtain W' in L :

$$W' = g_0 \cdot w_1 \cdot g_1 \cdot w_2 \cdot g_2 \cdot w_2 \cdot \dots$$

Splitting

Splitting lemma: If Z is an admissible path, W a word s.t. $\text{dist}(W,L) > h$, then W has more than h/m^2 Z -infeasible disjoint subwords. ($h=\varepsilon.n$)



Proof by contraposition:

W has less than $h'=h/m^2$ minimal Z -infeasible and disjoint subwords.

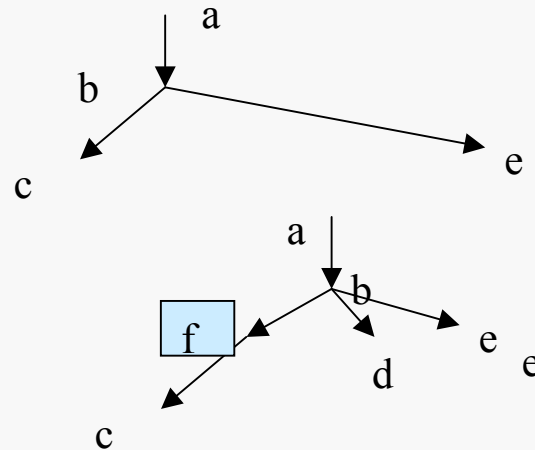
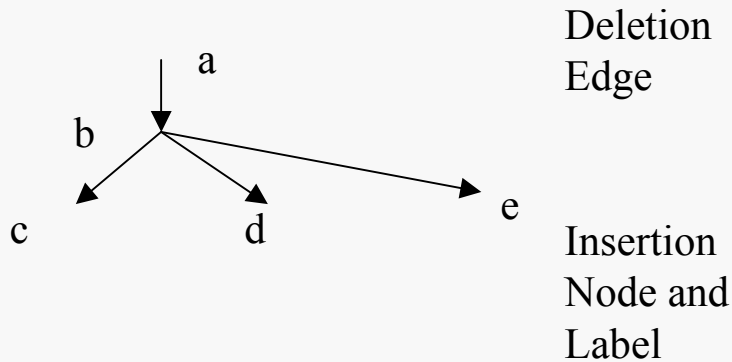
Removing the last letters provides a feasible cut F . $\text{Dist}(W,F) \leq h'$.

By the merging lemma $\text{Dis}(F,L) \leq m^2 h'$.

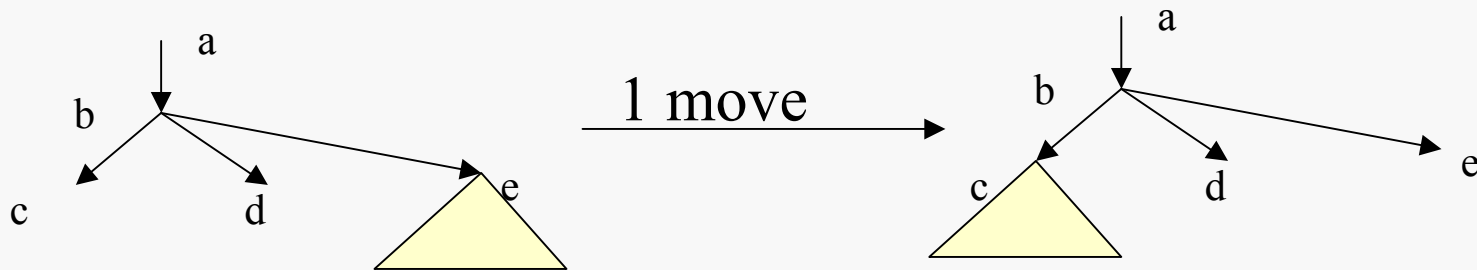
Hence $\text{Dist}(W,L) \leq h' + m^2 h'$

And $\text{Dist}(W,L) \leq h$

3b. Regular trees



Tree Edit distance with moves:

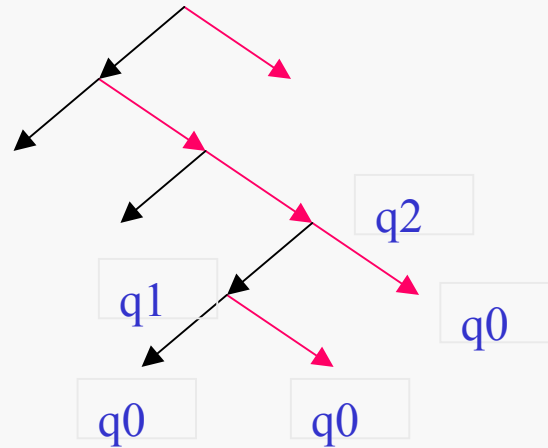
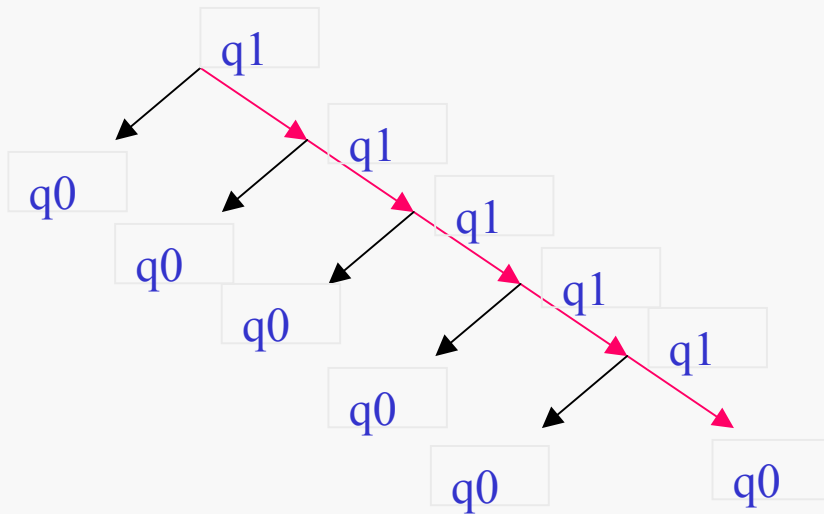


Distance Problem is NP-complete, non-approximable.

Tree automata

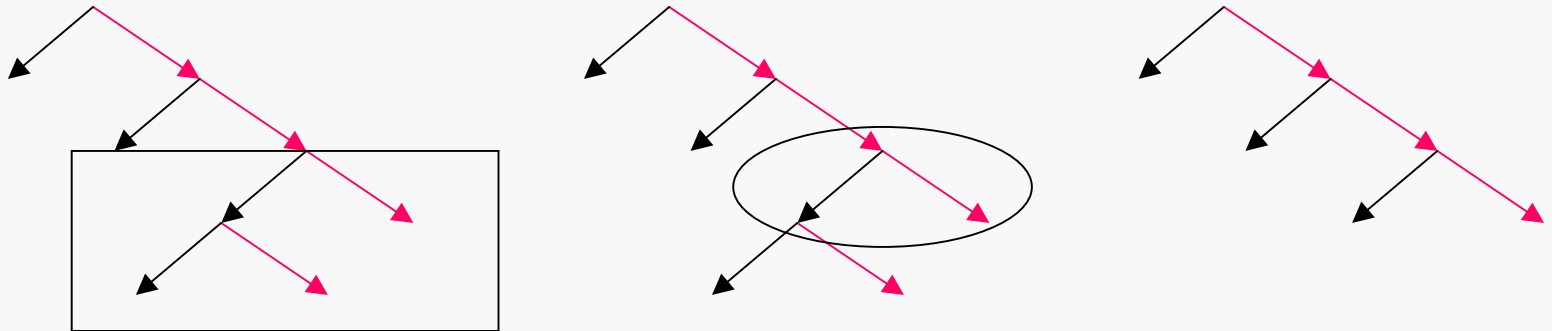
- $(q_0, q_0) \rightarrow q_1$
 - $(q_0, q_1) \rightarrow q_1$
- $A = (Q, q_0, \delta, q_1)$

- $(q_1, q_1) \rightarrow q_2$
- $(q_1, q_0) \rightarrow q_2$
- $(q_2, -) \rightarrow q_2$
- $(-, q_2) \rightarrow q_2$



Infeasible subtrees

Fact . If $\text{Distance}(T, L) > \varepsilon \cdot n$ then the number of infeasible subtrees of constant size is $O(n)$.



Tester for regular Trees

Tester. Input : T,A, ε

For $i \leq \frac{r^{2m+1}}{\varepsilon}$
Choose $N_i = \Theta\left(\frac{m \cdot r^{4m+3}}{\varepsilon^2}\right)$ random
nodes and subtrees t_j^i of size i

If all t_j^i of T are Z feasible, ACCEPT.

Theorem: Tester(T,A, ε) is an ε -tester for L(A).

Proof schema of the Tester

Theorem: Regular trees are testable.

Robustness lemma: If T is ε -far from L , then for every admissible path Z , there exists $i \leq \binom{r^{2m+1}}{\varepsilon}$ such that the number

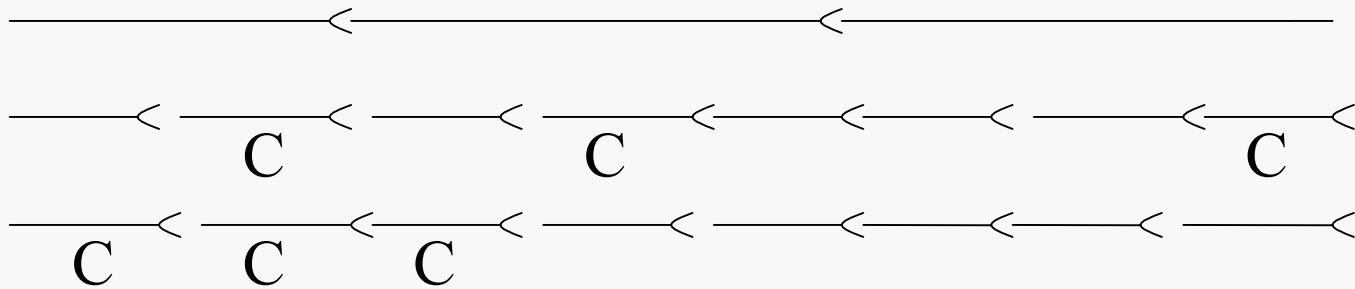
of Z -infeasible i -subtrees is at least $\frac{1}{r^{4m+3}} \cdot \varepsilon^2 \cdot n$.

Splitting lemma: if T is far from L there are many disjoint infeasible subtrees.

Amplifying lemma: If there are many infeasible subtrees, there are many small ones.

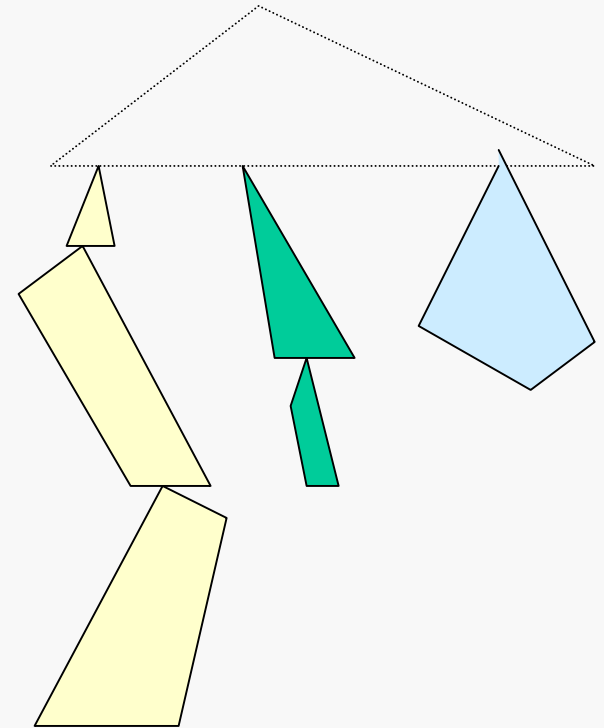
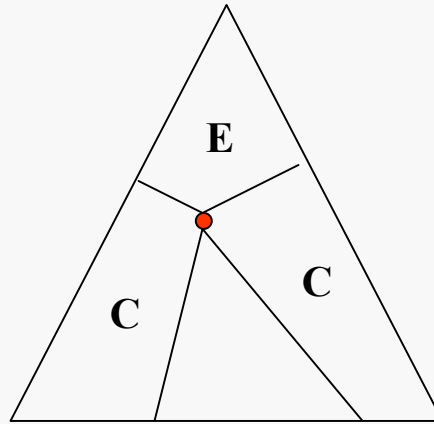
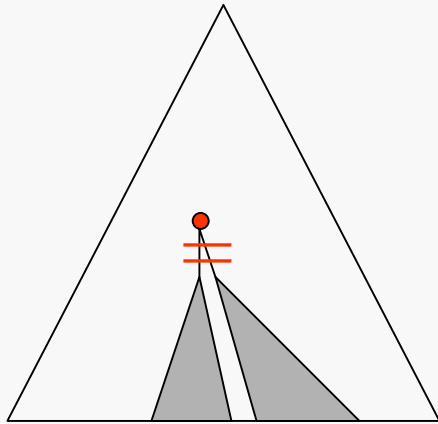
Splitting and Merging

Splitting and Merging on words:

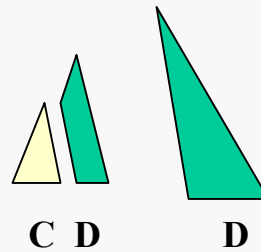
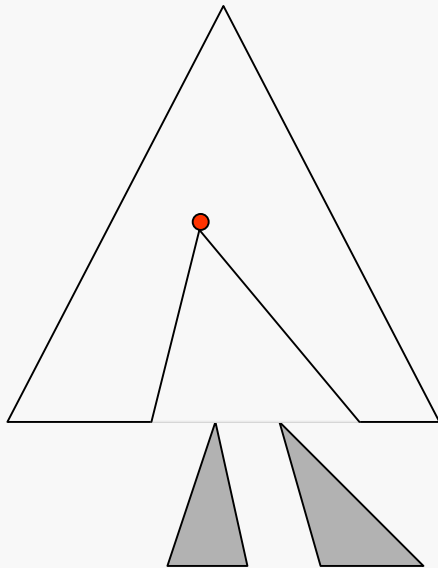


Splitting and Merging on trees:

Splitting and Merging trees



Connected Components



Corrected tree

4. Equivalent testing of Regular Languages

1. Inclusion

$L_1 \subseteq_\varepsilon L_2$ if $\forall v$ (except finitely) $v \in L_1 \Rightarrow v \in \text{close } L_2$

2. Equivalence

$L_1 \equiv_\varepsilon L_2$ if $L_1 \subseteq_\varepsilon L_2$ and $L_2 \subseteq_\varepsilon L_1$

Equivalence tester

If $L_1 = L_2$ then A accepts

If $\neg(L_1 \equiv_\varepsilon L_2)$ then A rejects with proba $\geq \frac{2}{3}$

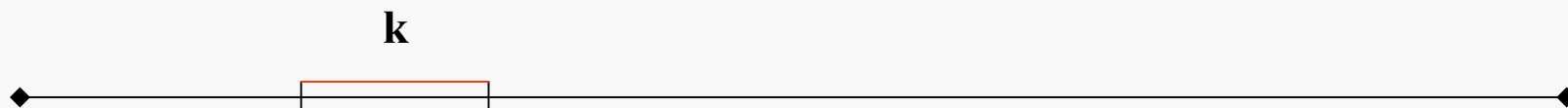
Statistics on words

Block statistics: b.stat

$$k = \frac{1}{\varepsilon}$$



Uniform statistics: u.stat



Construction of tester for regular languages exponential
in the size of the automaton

We need a construction polynomial in the size of the
automaton.

For equivalence testing, we use b.stat

Automata for Regular languages

Regular languages and automata

Non-deterministic automaton A , let A^k be the automaton accepting words of length k , reading v in Σ^k

Definition: v in Σ^k is an A^k loop if there are u, w such that

- Word $u.v.w$ is accepted by A^k
- State after u identical to the state after $u.v$

A finite set of loops is A^k -compatible if all loops can occur in an accepting word.

Definition: $H = \bigcup_{v_1, \dots, v_t \text{ } A^k\text{-loops}} \text{Convex-Hull}(b.stat(v_1), \dots, b.stat(v_t))$

Convex-hull: $\sum_{i=1, \dots, t} \lambda_i \times b.stat(v_i) \quad \text{s.t.} \quad \sum_i \lambda_i = 1$

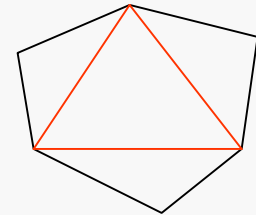
Automata for Regular languages

Basic property: $w \in L \Rightarrow w$ ε -close to $v.u_1 \dots u_l$ where $|u_i| \leq m$
 $\{u_1 \dots u_l\}$ is a multi-set of A^k -compatible loops

Proposition: $H = \bigcup_{v_1, \dots, v_t \text{ } A^k\text{-loops, } t=|\Sigma|^k+1, |v_i| \leq m} \text{Convex-Hull}(b.stat(v_1), \dots, b.stat(v_t))$

Caratheodory's theorem: in dimension d , convex hull of N points can be decomposed into in the union of convex hulls of $d+1$ points

Large loops can be decomposed.
Small loops (less than $m=|A|$) suffice.



Approximate Parikh mapping

Lemma: $\forall w \in L, \exists w', 0 \leq |w| - |w'| \leq \frac{m}{\varepsilon} \quad \text{dist}(w, w') \leq \frac{m}{\varepsilon}$
 $|b.stat(w) - b.stat(w')| \leq \frac{2m}{\varepsilon |w|}$ and $b.stat(w') \in H$

Find w'' ε close to w $w'' = v . u_1 \dots u_l$

Remove v , i.e. at most m block letters.

Lemma: For every X in H , for every n , there exists w in L s. t.

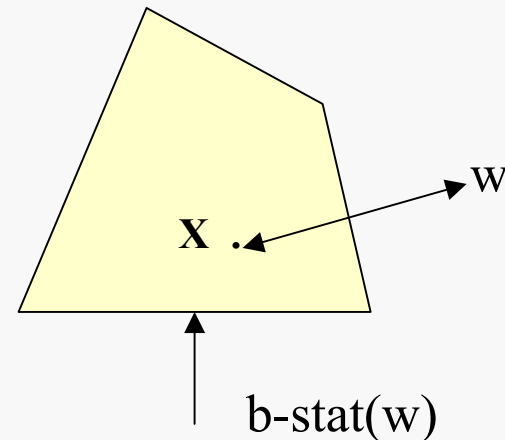
$$|w| \text{ close to } n \text{ and } |X - b.stat(w)| \leq \frac{c}{n}$$

Approximate Parikh mapping

Lemma: For every X in H , w in L s. t. $|X - b.stat(w)| \leq \delta$

$$\text{dist}(w, L) \leq \left(\frac{\delta}{2} + \varepsilon\right) \cdot n$$

H is a fair representation of L



Construction of H

Enumerate all loops: $|\Sigma|^{k.m}$

Number of b-stat is less : $m^{|\Sigma|^k}$

Some loops have same b-stat: ABBA and BBAA
#partitions of a word of length m with « big blocks »

Construct H by matrix iteration:

$$P_t = \left(\{b\text{-stat}(i \rightarrow^t j)\} \right) \quad P_{t+1} = P_1 \circ P_t$$

Construction of H

Lemma: compute a set I of at most $|\Sigma|^k + 1$ compatible loops,

$$|I| \leq m^{|\Sigma|^{k,m}} \quad H = \bigcup_{S \in I} \text{Convex-Hull}(S) \quad \text{time } O(m^{|\Sigma|^{k,m}})$$

Compute P_t for $t=1, \dots, m$

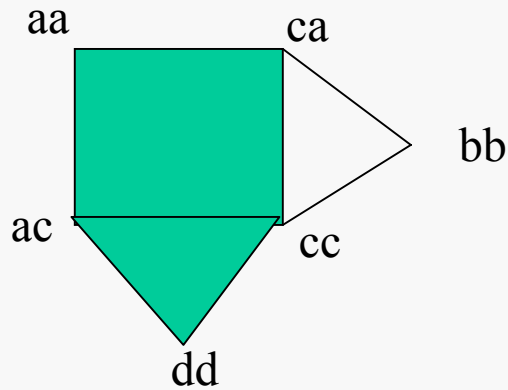
In the diagonals, find the b-stat of small loops, at most $m^{|\Sigma|^k}$
Consider subsets of at most $|\Sigma|^k + 1$ elements which are compatible.

Example

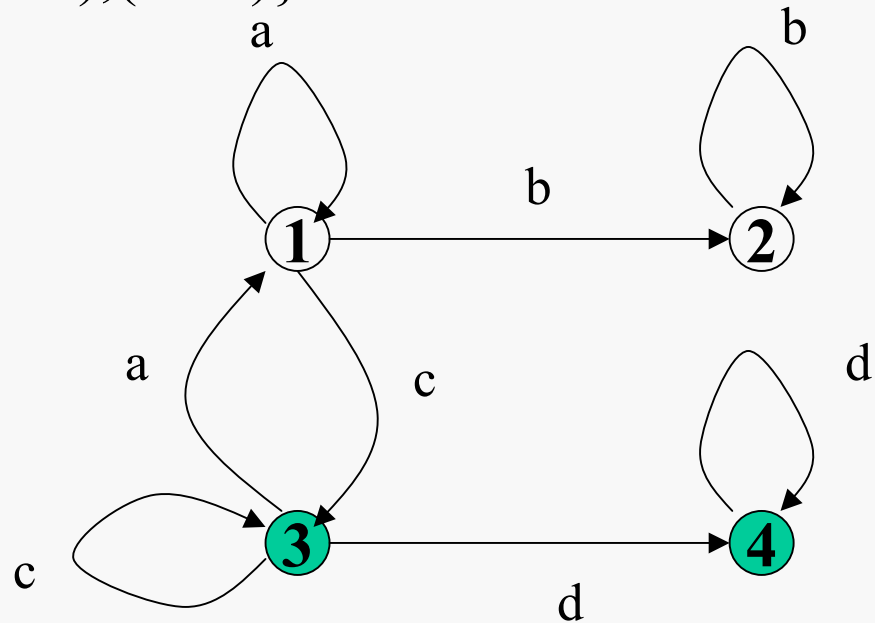
Automaton A:

Blocks, $k=2$, $m=4$, $|\Sigma|=4$, $|\Sigma|^k + 1 = 17$:

Loops: $\{(aa,ca:1),(bb,2),(cc,ac:3),(dd:4)\}$



H



A

Equivalence tester

Tester for w in L (regular):

Compute $b\text{-stat}(w)$ and H . Decide if $\text{dist}(w,L) > \epsilon.n$

Time is polynomial in $m=|L|$.

Previous tester was exponential in m .

Tester of $A \equiv_{\epsilon} B$

1. Compute H_A and H_B
2. Reject if H_A and H_B are different.

Time polynomial in $m=|A,B|$

Buchi Automata

Distance on infinite words:

Two words are ε -close if $\sup \lim_{n \rightarrow \infty} \text{dist}(w(n), w'(n)) \leq \varepsilon$

A word is ε -close to a language L if there exists w' in L s. t. W and w' are ε -close.

Statistics: set of accumulation points of $b.stat(w(n))_n$

H: compatible loops of connected components of accepting states

Tester for Buchi Automata:

Compute H_A and H_B

Reject if H_A and H_B are different.

Context-free Equivalence

Equivalence of CF grammars is undecidable,

Approximate equivalence in exponential.

Tester of $G_1 \equiv_\varepsilon G_2$

1. Compute H_A and H_B
2. Reject if H_A and H_B are different.

Applications:

Regular expression with squaring, negations....

Verification

Approaches:

1. Classical Model Checking: $M \models F$ (LTL, CTL,....)
formula. Check $M \models_{\epsilon} F$
2. Run of a protocol as a regular tree.

Direct application of the sampling techniques (on the states)

3. Protocols as a N-players games.

Property of the Nash equilibria. Mostly NP-hard.
Can it be approximated ?

3. Verification of probabilistic protocols.

Conclusion

- Verification is hard.
 - Approximate verification can be feasible.
1. Constant algorithm for Edit Distance with moves
 2. Testers and Correctors for regular words
 3. Tester for regular trees and corrector for regular trees
 4. Equivalence tester for automata (Buchi)