Quantitative Analysis of Security (with Probabilistic Automata)

Roberto Segala
University of Verona

Outline of Lecture

- Motivation
  - Can we use automata theory for security?
- Formal methods for security
  - Overview of techniques
- Probabilistic automata
  - Introduction to the model
- A case study
  - MAC1 protocol of Bellare and Rogaway
  - Approximated simulation relations
- Some open problems

Verification of Security Protocols

Our Question

- Can we use Probabilistic Automata?
  - Hierarchical verification
  - Compositional analysis
  - Simulation method
    - Local arguments to derive global properties
    - Rigorous proofs
  - Potentials for automatic verification
  - Potentials to draw connections to other areas

Nondeterminism and Probability

- Nondeterminism
  - User behavior (adversary in Dolev-Yao)
  - Relative speeds of agents
  - Agent behavior (usually deterministic)
  - Abstraction of details
- Probability
  - Users and agents flip coins
    - Nonces, keys, random protocols
- Quantitative analysis
  - Probability of attack (negligible)

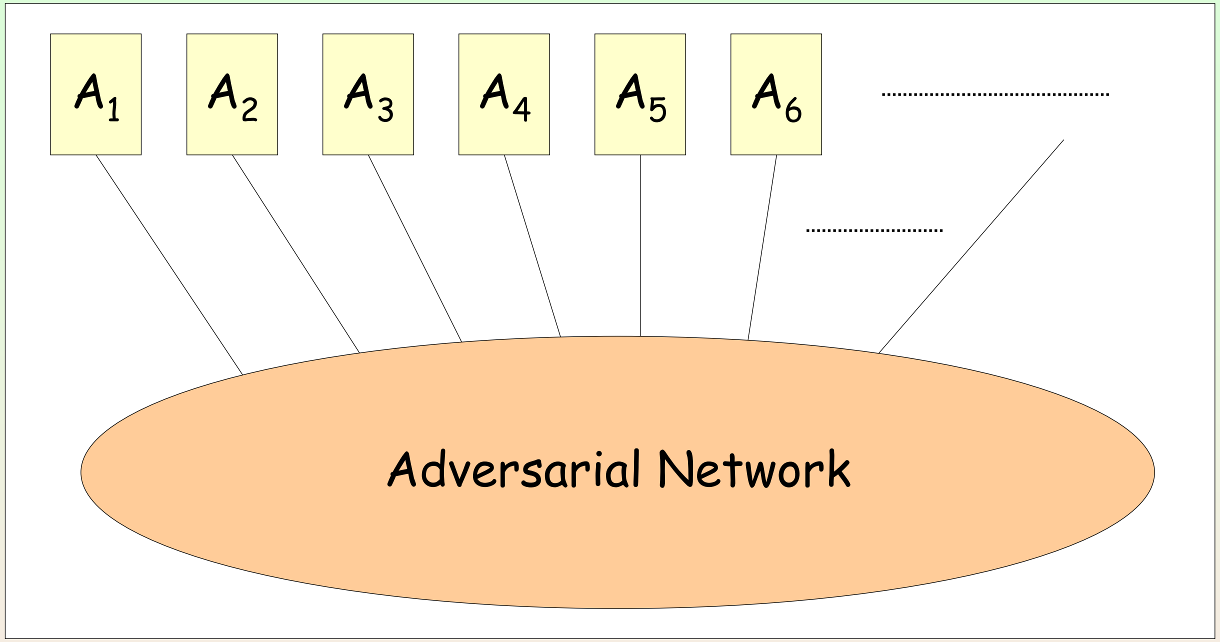
Formal Methods for Security: How?

- Provable security [GM84]
  - Based on Turing Machines (computational model)
  - Proofs by reduction to known difficult problems
- Dolev-Yao model [DY83]
  - Based on automata theory
  - Perfect cryptography
- Universally composable security [Can01]
  - Based on Interactive Turing Machines
  - Specification includes accepted attacks
- Reactive Simulatability [PW01]
  - Based on Probabilistic I/O Automata
  - Similar to UC framework

Provable Security

- Let h be a computationally hard function
- Let C be a cryptographic primitive
  - Collection of PPT algorithms that compute some functions
- State correctness of C as follows
  - There is no PPT algorithm A that computes some function f
- Prove correctness of C as follows
  - Suppose for the sake of contradiction that A exists
  - Build a PPT algorithm for h that uses A as a black box
  - This contradicts the hardness of h
- Correctness of C relies on hardness of h

Dolev-Yao Model



- Agents communicate through adversarial network
  - Network remembers everything
  - Network may block or reroute messages
  - Network may cast new messages

Dolev-Yao Model: Assumptions

- Symbolic (typical use of the model)
  - Messages are symbols
  - Cryptography is perfect
  - Adversary power limited by a deduction system
    - Nonces are always fresh
    - No ability to decrypt without decryption key
  - Adversary is nondeterministic

- Computational
  - Messages are bit strings
  - Adversary governed by PPT functions

Symbolic Dolev-Yao Model

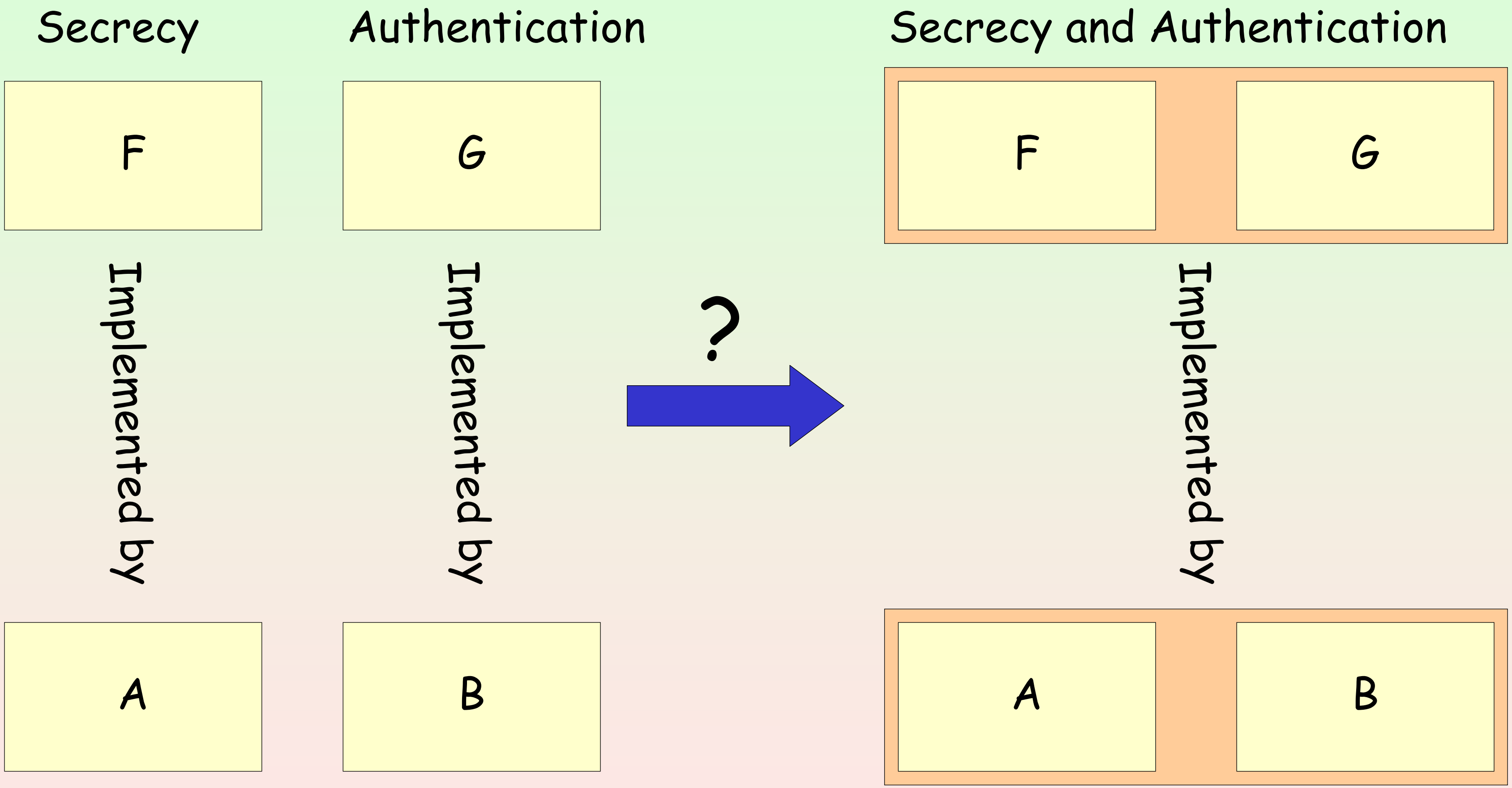
- Analysis is simple
  - The system is described by an automaton
  - Show that no path leads to failure or attack
  - Plenty of techniques from concurrency theory
    - Invariants
    - Compositional analysis
    - Language properties
    - Model checking
- Sound with respect to computational [AR00]
  - Attack in computational model yields attack in symbolic model
  - Need some assumptions on underlying cryptoprimitives
    - Non malleability

Symbolic Dolev-Yao Deductions

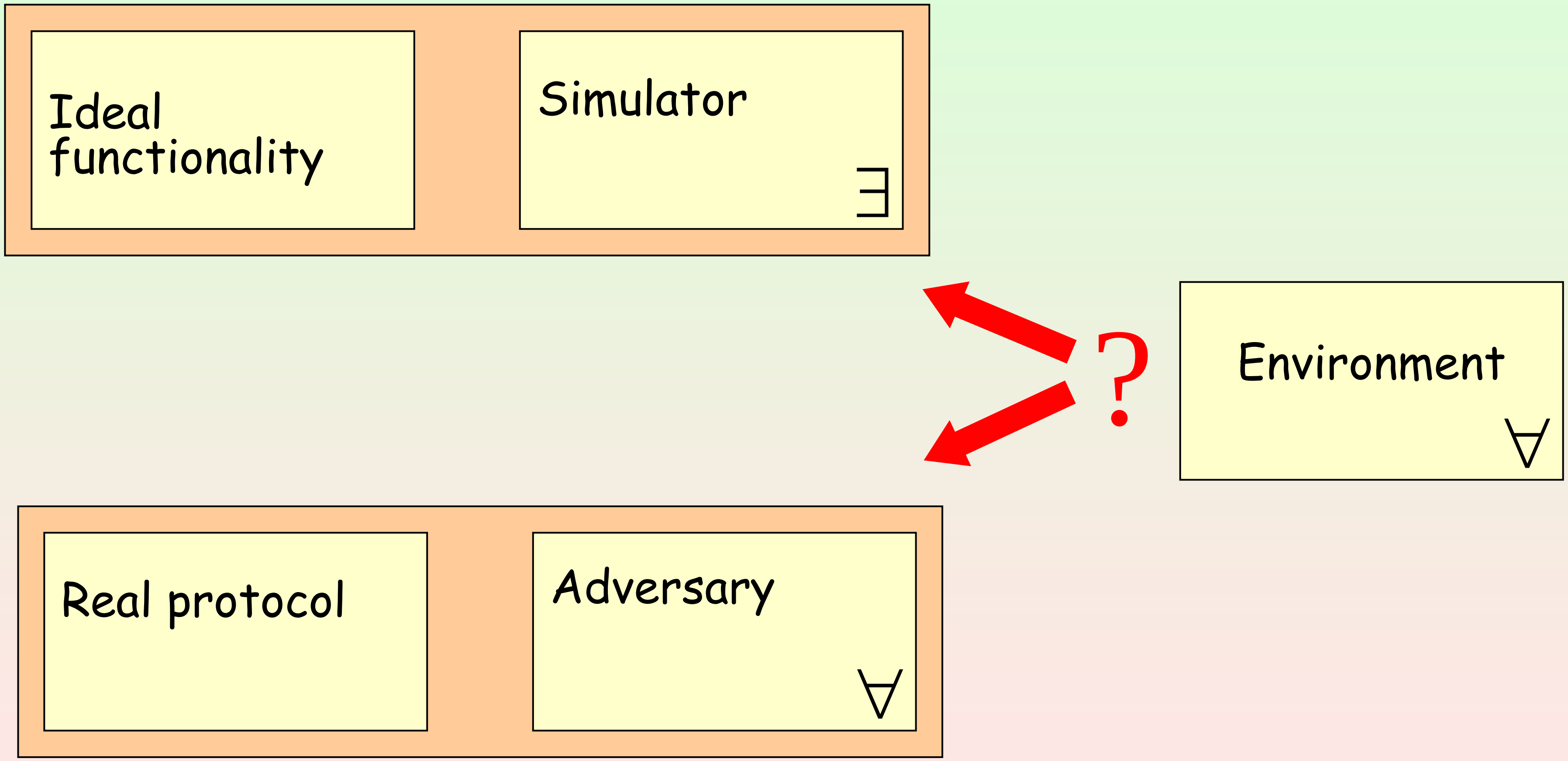
- $A \mid\!- X, \quad A \mid\!- Y \quad \Rightarrow A \mid\!- (X,Y)$
- $A \mid\!- (X,Y) \quad \Rightarrow A \mid\!- X$
- $A \mid\!- (X,Y) \quad \Rightarrow A \mid\!- Y$
- $A \mid\!- X, \quad A \mid\!- k \quad \Rightarrow A \mid\!- \{X\}_k$
- $A \mid\!- \{X\}_k, \quad A \mid\!- k \quad \Rightarrow A \mid\!- X$

- Automaton transitions
  - Agents add messages to adversary
  - Adversary casts messages according to deductions
- Invariants
  - Signature deducible only if it exists already
- Property
  - Answers always generated by correct agents

UC [Can01] and RSim [PW01] Motivation



UC-Framework [Canetti]



Reactive Simulatability [Pfitzmann Waidner]

- Similar to UC Framework
- Based on PIOAs rather than ITMs
- More elaborated on verification techniques
- Large collection of definitions
  - Crypto library [BPW03]

Fine, but how do we prove Facts?

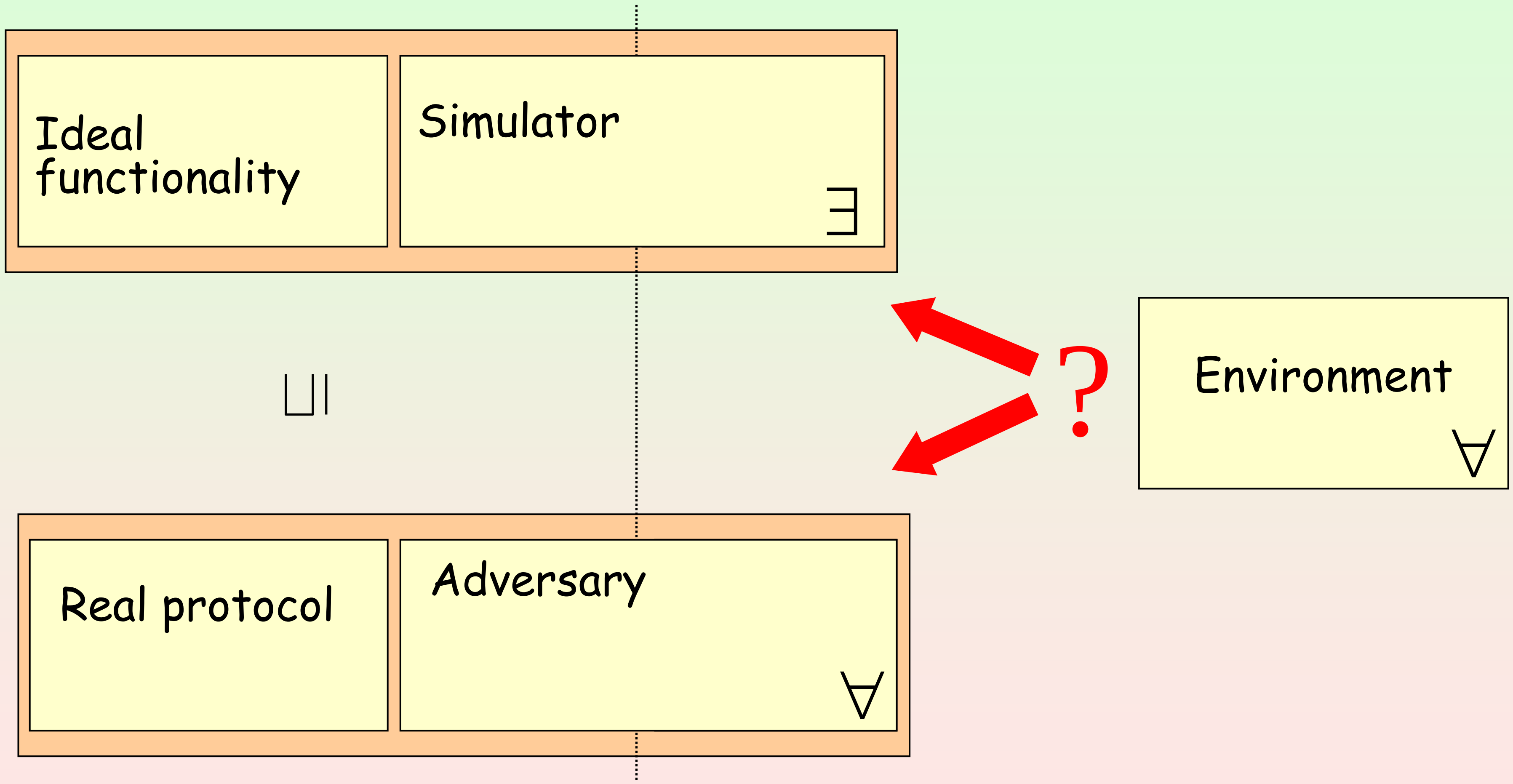
- Provable security
  - Semi-formal arguments
  - A lot of wording
- Dolev-Yao
  - Semi-formal arguments
  - … or typical arguments from concurrency theory
- UC Framework
  - Semi-formal arguments
- Reactive simulatability
  - Semi-formal arguments
  - "Simulation" up to "error sets"
  - Negligible probability of error sets

Can we be More Rigorous?

- Use Dolev-Yao and Soundness
  - Concurrency theory has plenty of techniques
- Use Process Algebraic formalisms [MRST06 and earlier]
  - Expressions denote PPT computable functions
  - Equivalence denotes indistinguishability
  - Axiomatic reasoning
- Use game transformations [Sho04,Bla05]
  - Correctness in provable security expressed as a game
  - Transform games preserving correctness
- Use Automata Theory [CCKLLPS06,ST07]
  - Add computational assumptions
  - Extend known techniques (simulation method)

UC-Security with PIOAs

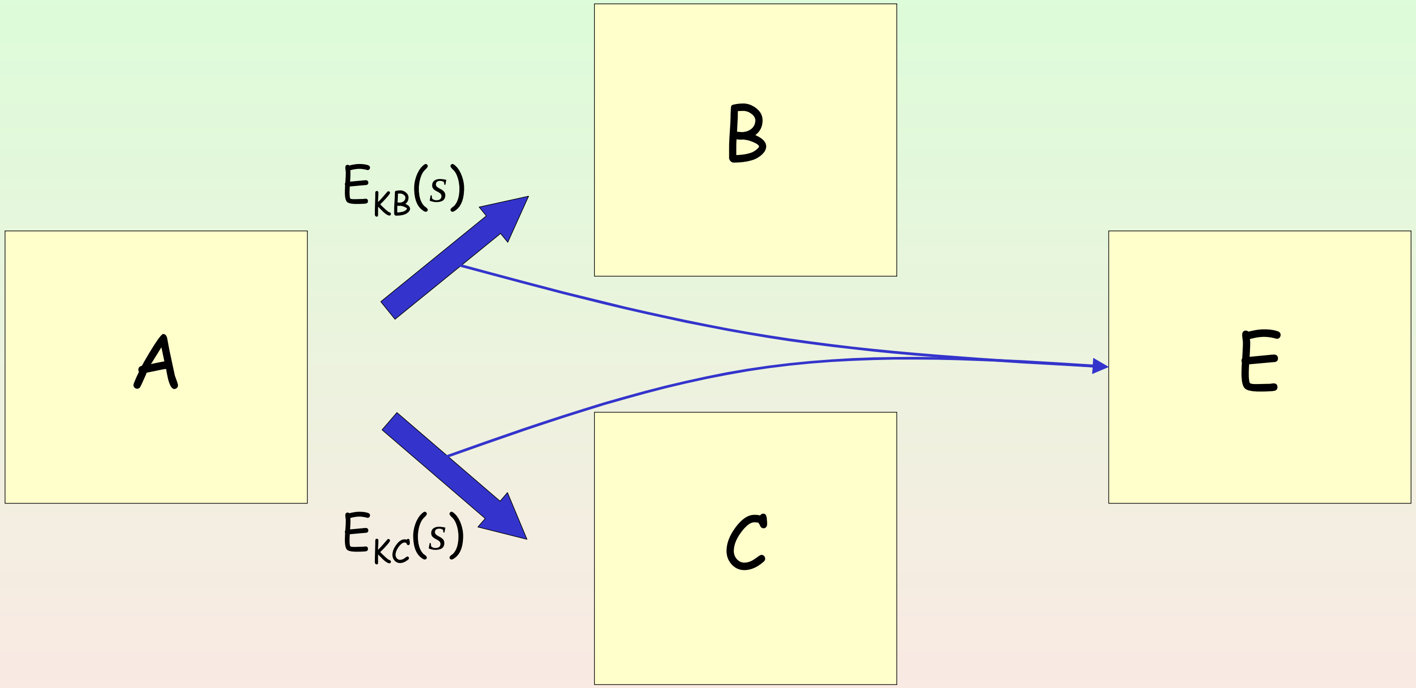
[Canetti, Cheung, Kaynar, Liskov, Lynch, Pereira, Segala]



Nondeterminism: why There?

- If we have several components
  - Who moves first (nondeterminism)?
  - Can the order of operations reveal secrets?
- If we expect input
  - What input do we receive?
- If we have partial specification
  - How do we implement (nondeterminism)?
- Nondeterminism resolved by a "scheduler"
  - Not all resolutions are safe

Example of Nondeterminism



- Order of messages may reveal one bit of s to E

Approaches to Nondeterminism

- UC framework
  - ITMs have a token passing mechanism
  - No nondeterminism
- Reactive simulatability
  - Again token passing mechanism
  - Nondeterminism based on local information only
- Process Algebras
  - Scheduler sees only enabled action type
- Task PIOAs
  - Define equivalence classes of states and actions
  - Scheduler sees only equivalence classes, not elements
- Symbolic Dolev-Yao
  - No probability
  - Symbols hide information
- Careful specifications
  - Avoid dangerous nondeteminism in the specification
  - Is it always possible?

….. So …….

let's concentrate on …

Automata

Automata

$$A = (Q\ ,\ q_0\ ,\ E\ ,\ H\ ,\ D)$$

Transition relation
$D \subseteq Q \times (E \cup H) \times Q$

Internal (hidden) actions

External actions: $E \cap H = \emptyset$

Initial state: $q_0 \in Q$

States

Probabilistic Automata

$$PA = (Q\ ,\ q_0\ ,\ E\ ,\ H\ ,\ D)$$

Transition relation

$D \subseteq Q \times (E \cup H) \times \mathrm{Disc}(Q)$

Internal (hidden) actions

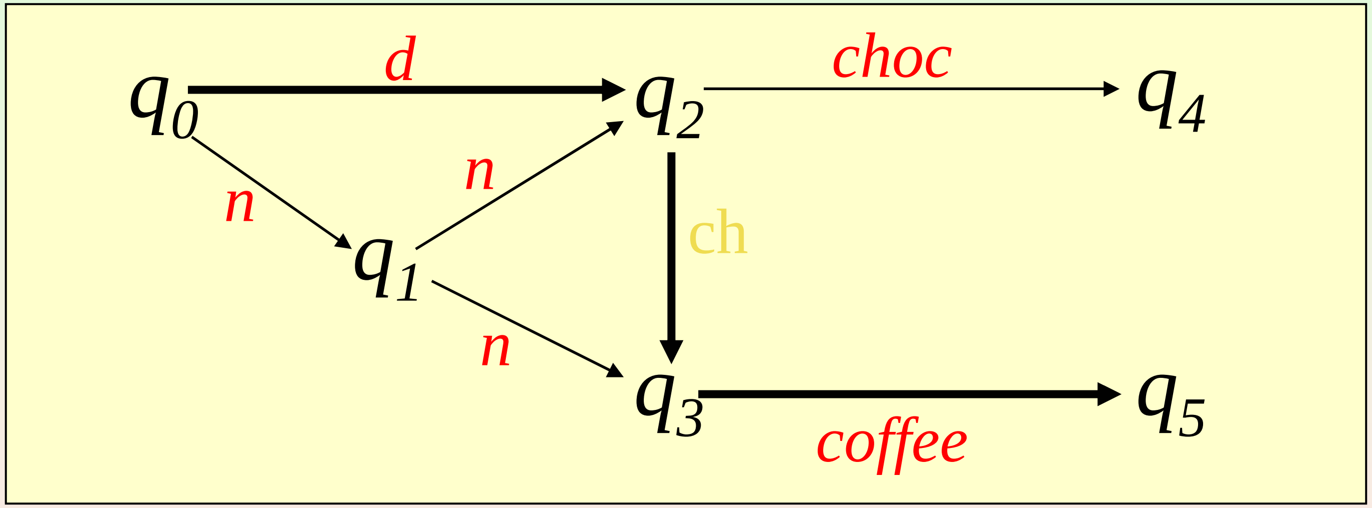
External actions: $E \cap H = \emptyset$

Initial state: $q_0 \in Q$

States

Example: Automata

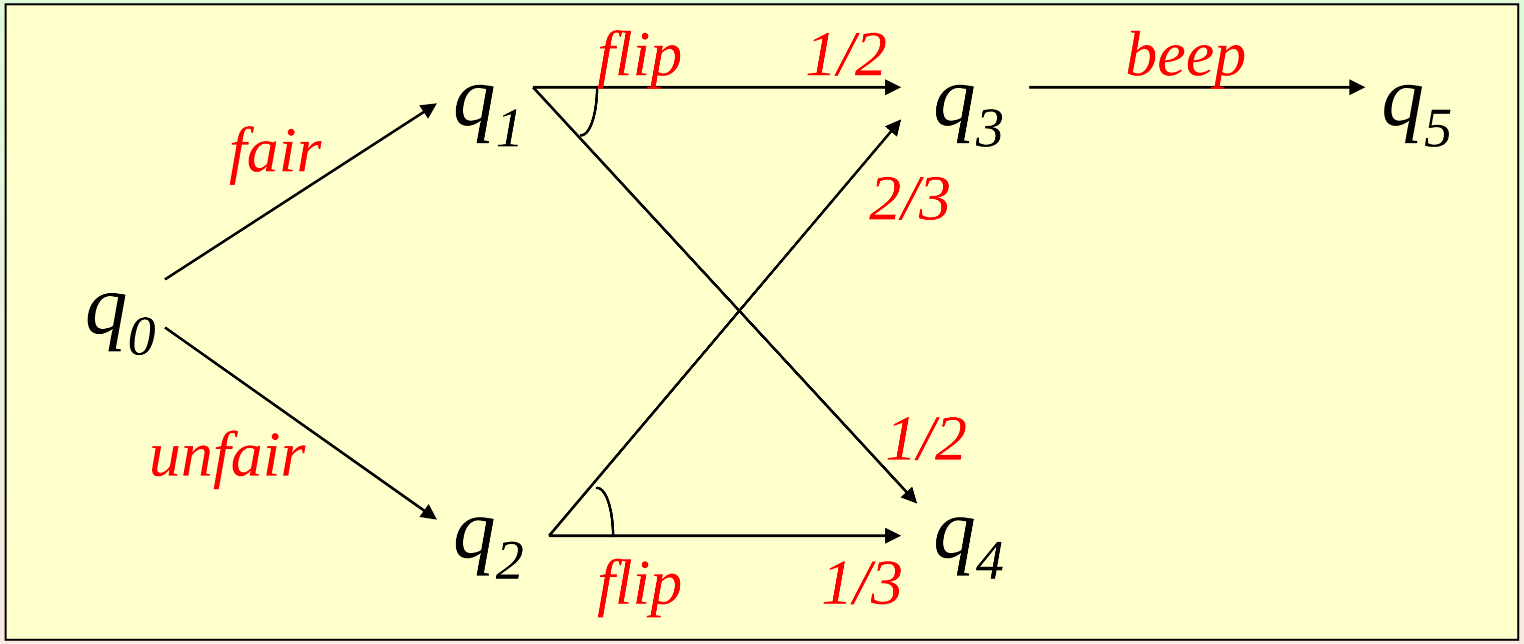
$$A = (Q, q_0, E, H, D)$$



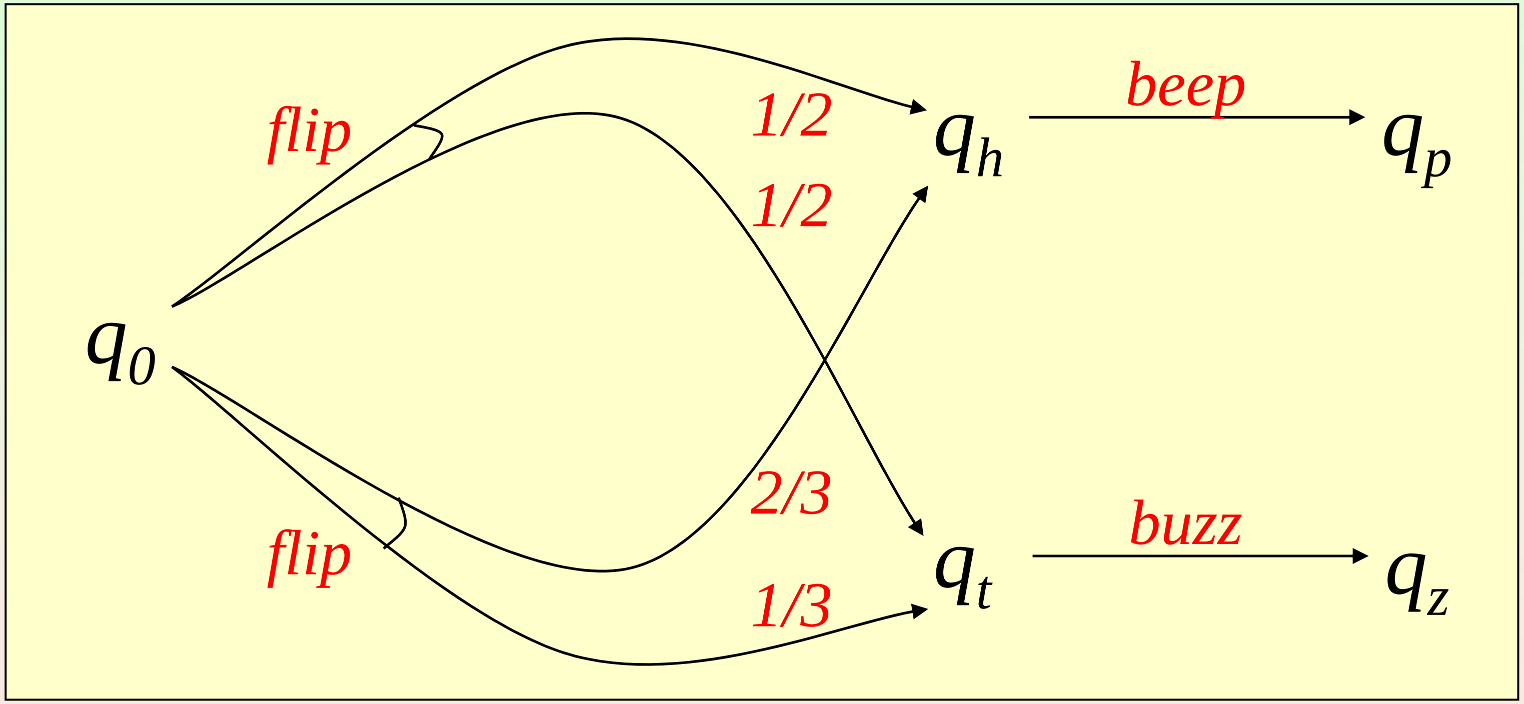
Execution: $q_0\ n\ q_1\ n\ q_2\ ch\ q_3\ coffee\ q_5$

Trace: $n\ n\ coffee$

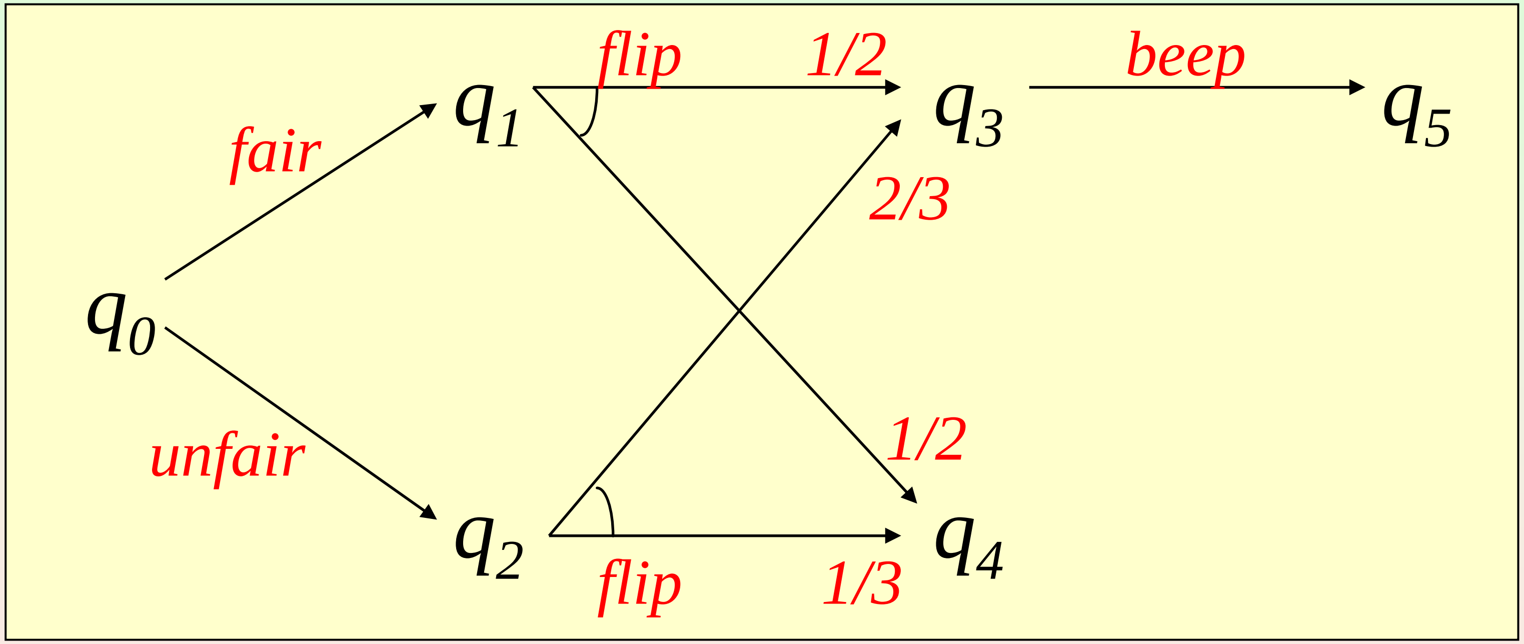
Example: Probabilistic Automata



Example: Probabilistic Automata

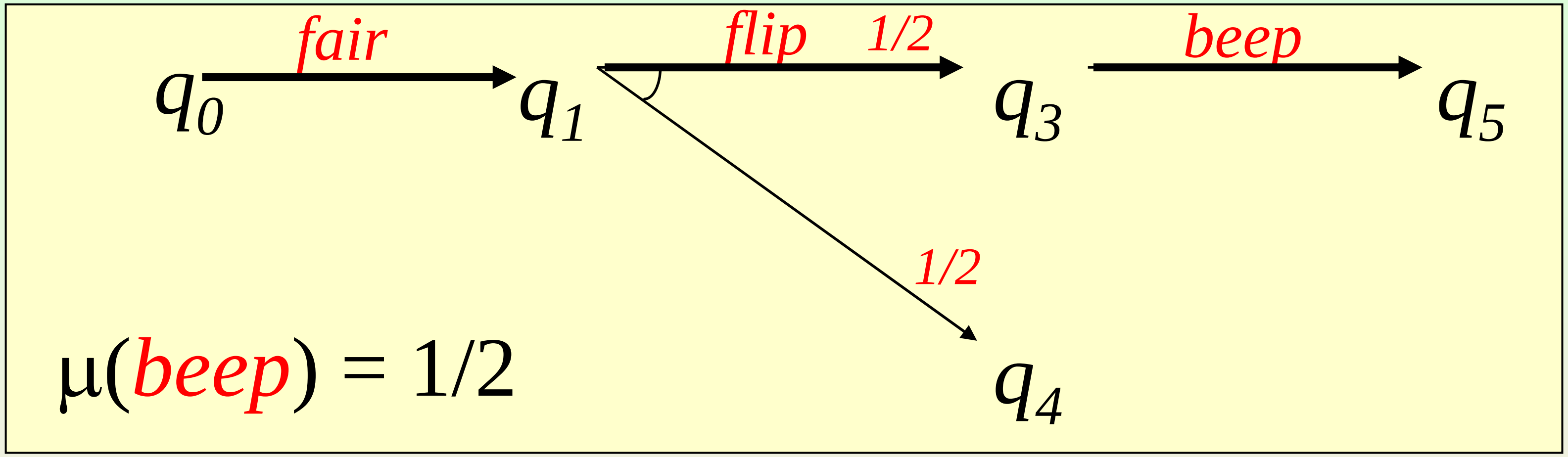


Example: Probabilistic Automata

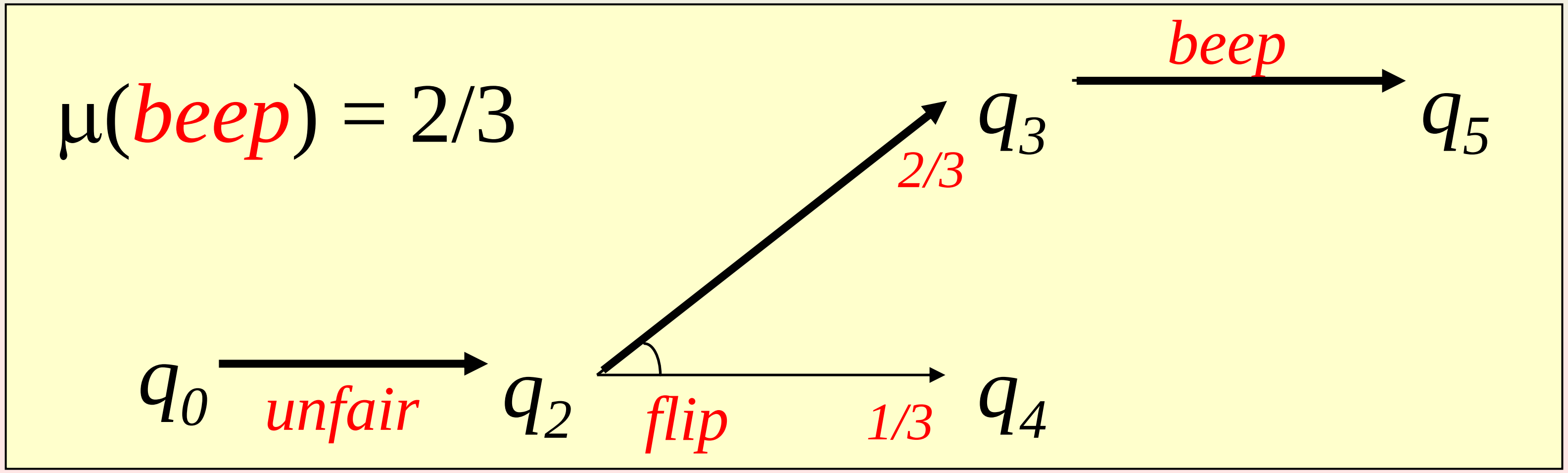


What is the probability of beeping?

Example: Probabilistic Executions



1/2



2/3

Example: Probabilistic Executions

Measure Theory

- Sample set
  - Set of objects Ω
- Sigma-field (σ-field)
  - Subset F of 2^Ω satisfying
    - Inclusion of Ω
    - Closure under complement
    - Closure under countable union
    - Closure under countable intersection
- Measure on (Ω, F)
  - Function μ from F to $\Re^{\geq 0}$
    - For each countable collection $\{X_i\}_I$ of pairwise disjoint sets of F, $\mu(\cup_I X_i) = \Sigma_I \mu(X_i)$
- (Sub-)probability measure
  - Measure μ such that $\mu(\Omega) = 1$ ($\mu(\Omega) \leq 1$)
- Sigma-field generated by $C \subseteq 2^\Omega$
  - Smallest σ-field that includes C

Example: set of executions

Study probabilities of sets of executions

which sets can I measure?

Why not $F = 2^\Omega$?

Flip a fair coin infinitely many times

$\Omega = \{h,t\}^\infty$

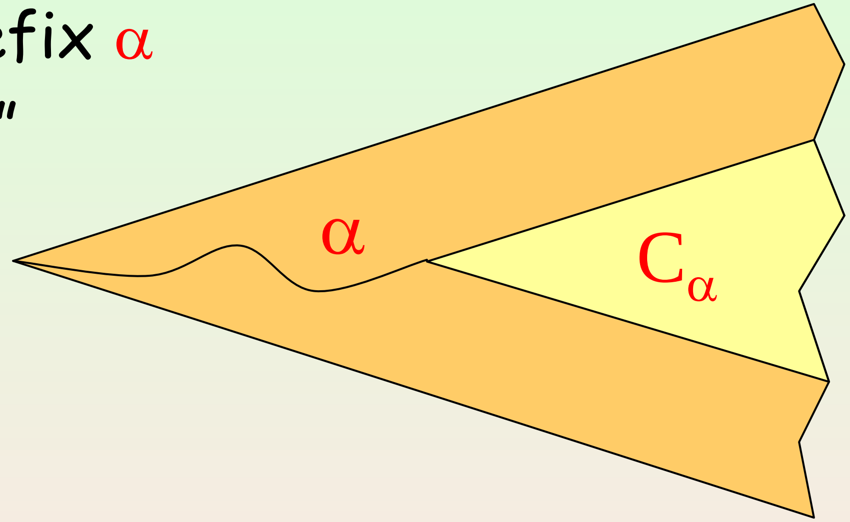
$\mu(\omega) = 0$ for each $\omega \in \Omega$

$\mu(\text{first coin } h) = 1/2$

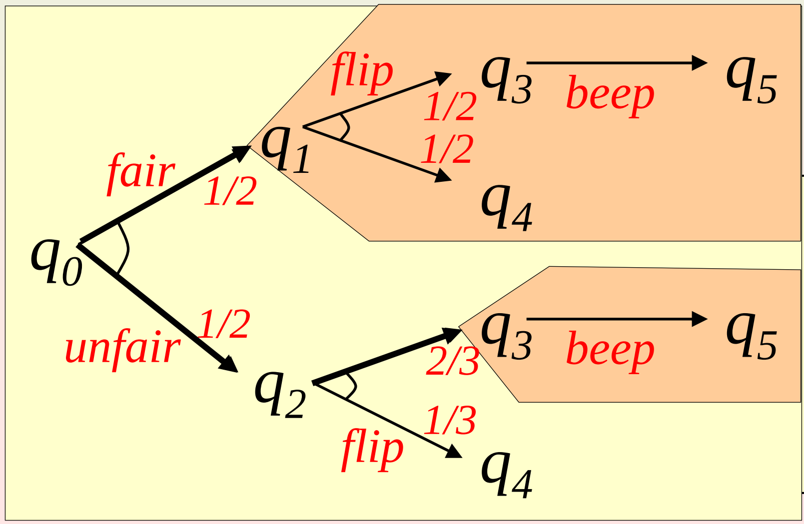
Theorem: there is no σ-additive function μ on 2^Ω such that
- $\mu(\omega) = 0$ for each $\omega \in \Omega$, and
- $\mu(\Omega) > 0$.

Cones and Measures

- Cone of α
  - Set of executions with prefix α
  - Represent event "α occurs"



- Measure of a cone
  - Product edges of α



extends uniquely

σ-field generated by cones

Examples of Events

- Eventually action a occurs
  - Union of cones where action a occurs once
- Action a occurs at least n times
  - Union of cones where action a occurs n times
- Action a occurs at most n times
  - Complement of action a occurs at least n+1 times
- Action a occurs exactly n times
  - Intersection of previous two events
- Action a occurs infinitely many times
  - Intersection of action a occurs at least n times for all n
- Execution α occurs and nothing is scheduled after
  - Set consisting of α only
  - C_α intersected complement of cones that extend α

Schedulers - Probabilistic Executions

Scheduler

Function $\sigma : exec^*(A) \rightarrow SubDisc(D)$

if $\sigma(\alpha)((q,a,\nu)) > 0$ then $q = lstate(\alpha)$

Probabilistic execution generated by σ from state r

Measure $\mu_{\sigma,r}$

$$\mu_{\sigma,r}(C_s) = 0 \quad \text{if } r \neq s$$

$$\mu_{\sigma,r}(C_r) = 1$$

$$\mu_{\sigma,r}(C_{\alpha aq}) = \mu_{\sigma,r}(C_\alpha) \cdot \left(\sum_{(s,a,\nu)\in D} \sigma(\alpha)((s,a,\nu))\nu(q) \right)$$

Other Models

- Reactive and generative systems
  - Restricted forms of transitions
- Labeled Concurrent Markov Chains
  - Restricted forms of transitions
- Rabin's Probabilistic Automata
  - Introduced in the context of language theory
  - Extended by our Probabilistic Automata
- Unlabeled systems [Var85,BA95,BK98]
  - Can be Probabilistic Automata with a single invisible action
  - Labels may be associated with states
  - The theory does not change
- Markov Chains
  - Unlabeled systems that enable one transition from each state
- Probabilistic Input/Output Automata
  - Add Input/Output distinction on actions
  - Useful to handle composition of generative PAs

Composition of Probabilistic Automata

$A_1 = (Q_1,q_1,E_1,H_1,D_1)$ $\quad\parallel\quad$ $A_2 = (Q_2,q_2,E_2,H_2,D_2)$

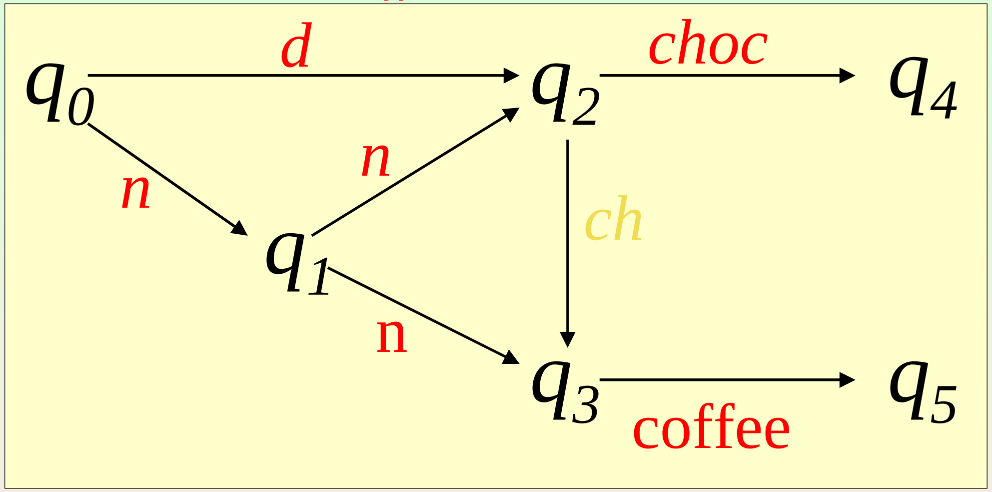
$$A_1 \parallel A_2 = (Q_1 \times Q_2 , (q_1,q_2) , E_1 \cup E_2 , H_1 \cup H_2 , D)$$

$$D = \left\{ (q,a,(s_1,s_2)) \;\middle|\; \begin{array}{l} \text{if } a \in E_i \cup H_i \text{ then } (\pi_i(q) , a , s_i) \in D_i \\ \text{if } a \notin E_i \cup H_i \text{ then } s_i = \pi_i(q) \end{array} \quad i \in \{1,2\} \right\}$$

$$D = \left\{ (q,a,\mu_1 \times \mu_2) \;\middle|\; \begin{array}{l} \text{if } a \in E_i \cup H_i \text{ then } (\pi_i(q) , a , \mu_i) \in D_i \\ \text{if } a \notin E_i \cup H_i \text{ then } \mu_i = \delta(\pi_i(q)) \end{array} \quad i \in \{1,2\} \right\}$$

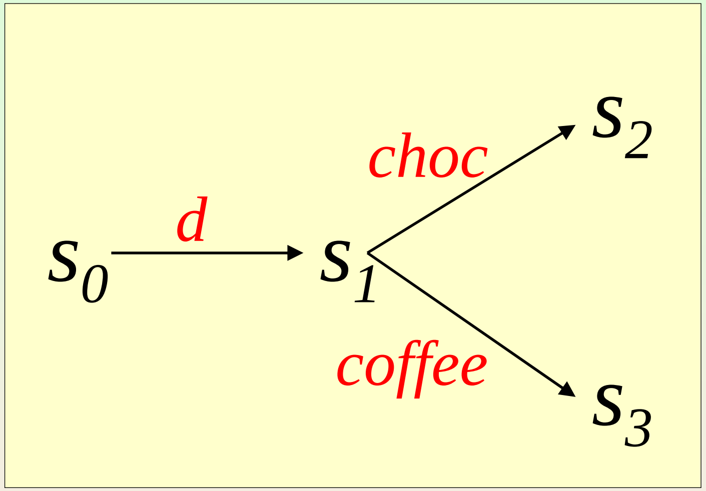
Example: Composition of Automata

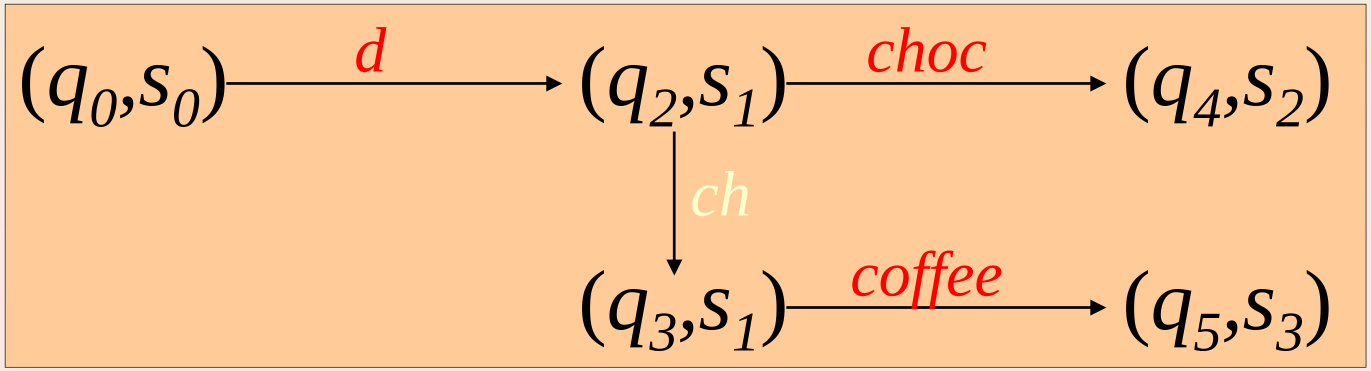
$E = \{n, d, choc, coffee\}$



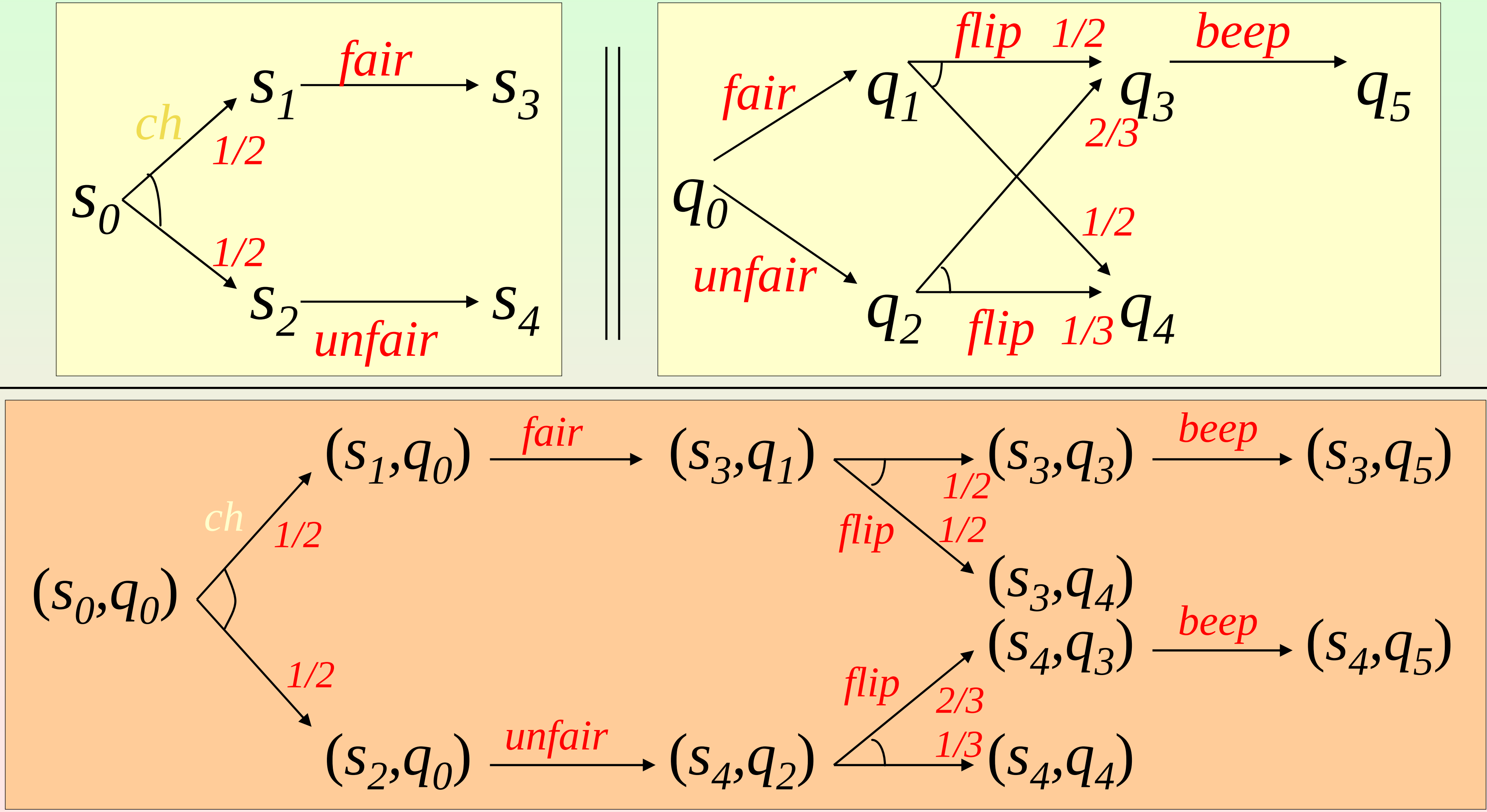
$\|$

$E = \{n, d, choc, coffee\}$





Ex. Composition of Probabilistic Automata



Projections

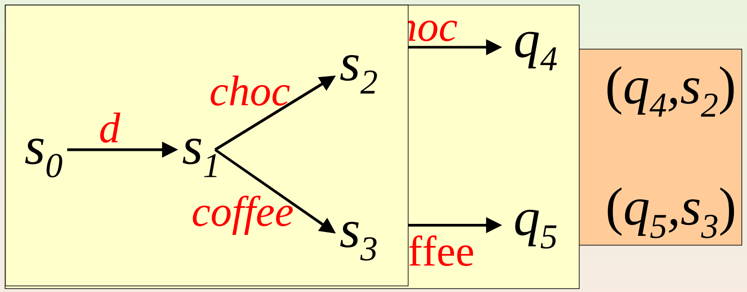
Let α be an execution of $A_1 \parallel A_2$

$\alpha = (q_0,s_0)\ d\ (q_2,s_1)\ ch\ (q_3,s_1)\ coffee\ (q_5,s_3)$

What are the contributions of A_1 and A_2?

$\pi_1(\alpha) \equiv q_0\ d\ q_2\ ch\ q_3\ coffee\ q_5$

$\pi_2(\alpha) \equiv s_0\ d\ s_1\ coffee\ s_3$



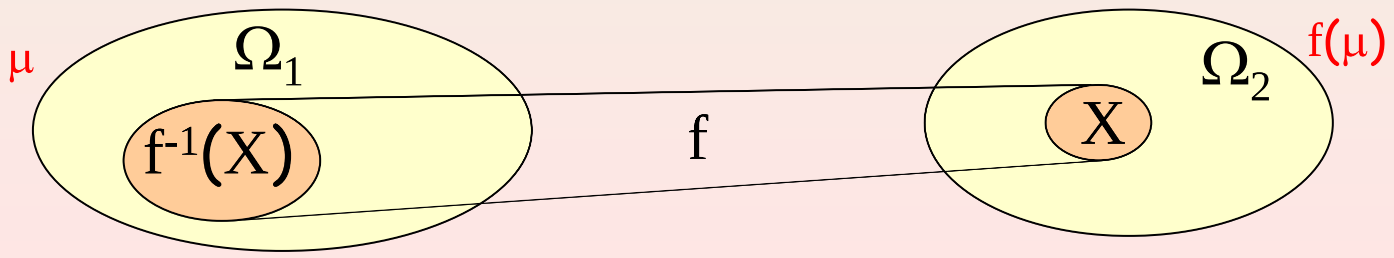
Theorem

$\alpha \in execs(A_1||A_2)$ iff $\forall_{i \in \{1,2\}}\ \pi_i(\alpha) \in execs(A_i)$

Measure Theory: Image Measure

- Measurable function from (Ω_1, F_1) to (Ω_2, F_2)
  - Function f from Ω_1 to Ω_2
  - For each element X of F_2, $f^{-1}(X) \in F_1$

- Image measure $f(\mu)$
  - $f(\mu)(X) = \mu(f^{-1}(X))$



Projections

The projection function is measurable

$\pi(\mu)$: image measure under π of μ

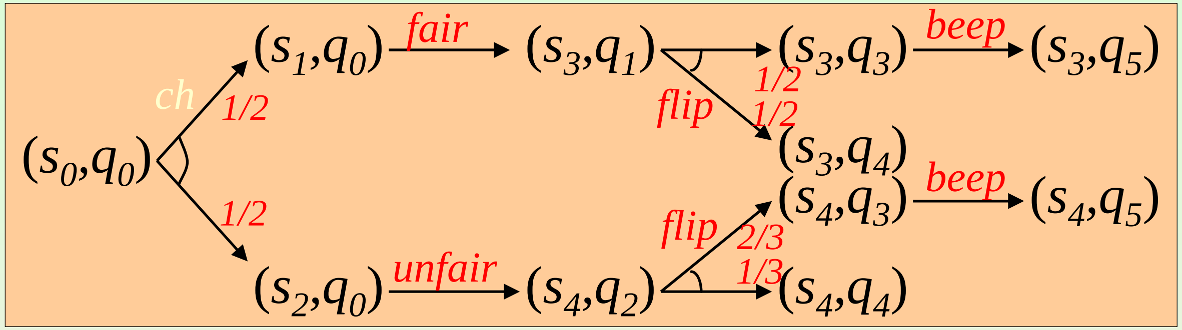
Theorem

If μ is a probabilistic execution of $A_1 \parallel A_2$

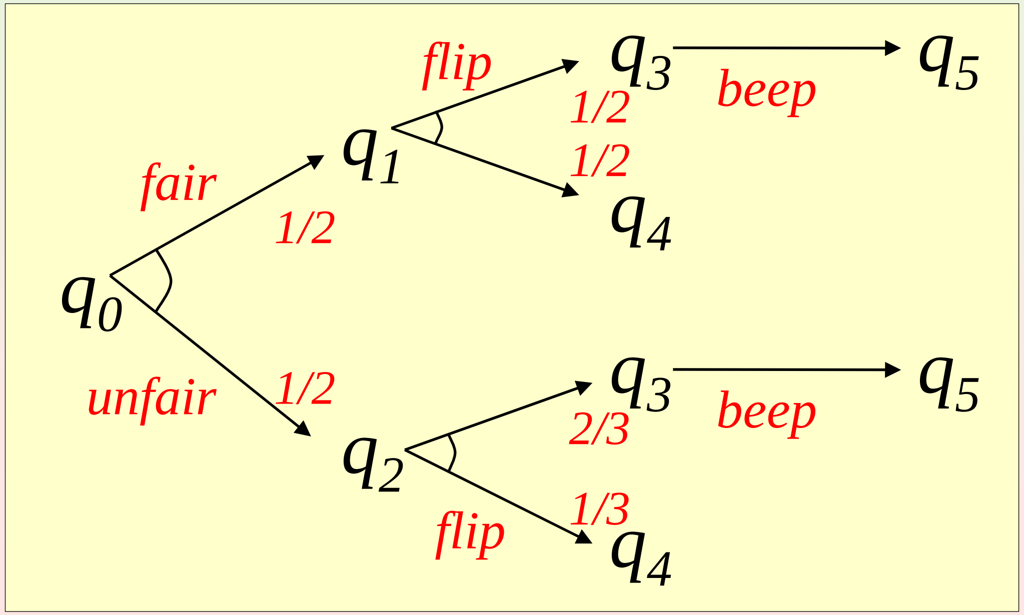
then

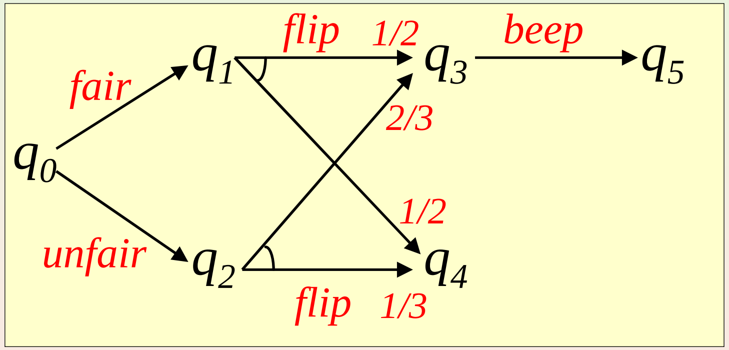
$\pi_i(\mu)$ is a probabilistic execution of A_i

Example: Projection



Projection onto right component





Note that the scheduler is randomized

Trace Distributions

The *trace* function is measurable

Trace distribution of μ

$tdist(\mu)$: image measure under *trace* of μ

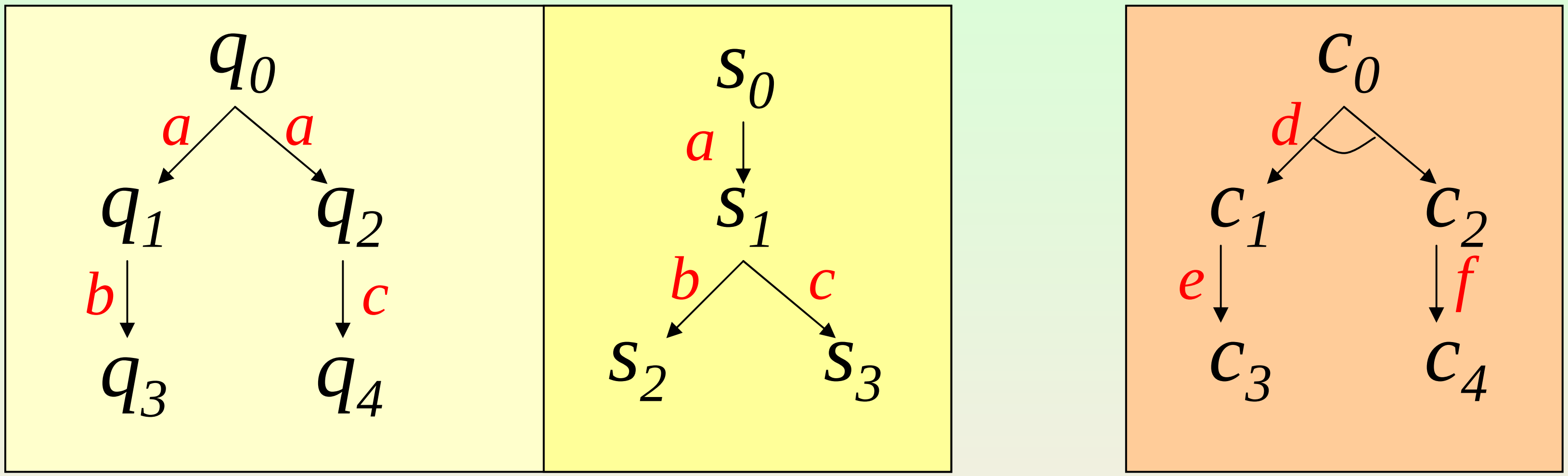
Trace distribution inclusion preorder

$A_1 \leq_{\mathrm{TD}} A_2$ iff $tdists(A_1) \subseteq tdists(A_2)$

Summing Up

Automata		Probabilistic Automata
⇓	schedulers	⇓
Executions		Probabilistic Executions (measures over executions)
⇓	trace function	⇓
Traces		Trace distributions (measures over traces)
⇓	implementation relation	⇓
Trace inclusion		Trace distribution inclusion

Trace Distribution Inclusion is not Compositional



Solution: close under all contexts

$$(s_0,c_0) \xrightarrow{a} (s_1,c_0) \xrightarrow{d} (s_1,c_1) \xrightarrow{e} (s_1,c_3) \xrightarrow{b} (s_2,c_3)$$

$$(s_1,c_0) \xrightarrow{d} (s_1,c_2) \xrightarrow{f} (s_1,c_4) \xrightarrow{c} (s_3,c_4)$$

Trace distribution precongruence

$A \le_{TDC} B$ iff $\forall C \; A||C \le_{TD1} B||C$

Quantitative Extension of Trace Distribution Inclusion

- $A \leq B$ iff $\forall C$
  - If ν is a trace distribution of A||C, then
  - There exists a trace distribution ν' of B||C
  - Such that ν and ν' are PPT indistinguishable

- Technical detail
  - Need to parameterize PAs by security value k
  - Need to ensure PAs are PPT constructable

… yet, Proving Language Inclusion is Difficult

- Language inclusion is a global property
  - Need to see the whole result of resolving nondeterminism
- We seek local proof techniques
  - Local arguments are easier
- We use simulation relations

Strong Bisimulation on Automata

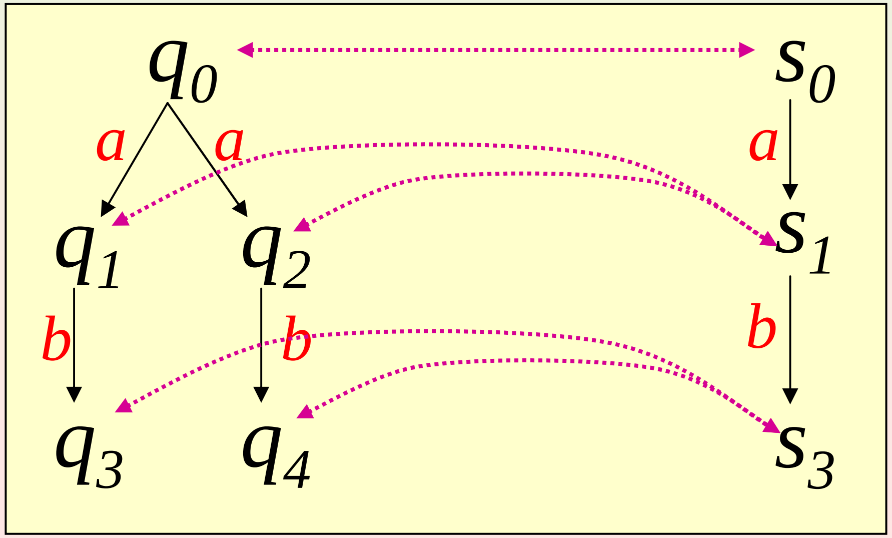
Strong bisimulation between A_1 and A_2

Relation $\boldsymbol{R} \subseteq Q \times Q$,

$Q = Q_1 \uplus Q_2$, such that

$\forall\, q, s, a, q' \;\exists\, s'$

$$\begin{tikzcd} q \arrow[r, "a"] \arrow[d, dash, dotted, "\boldsymbol{R}"'] & q' \arrow[d, dash, dotted, "\boldsymbol{R}"] \\ s \arrow[r, "a"] & s' \end{tikzcd}$$



Strong Bisimulation on Probabilistic Automata

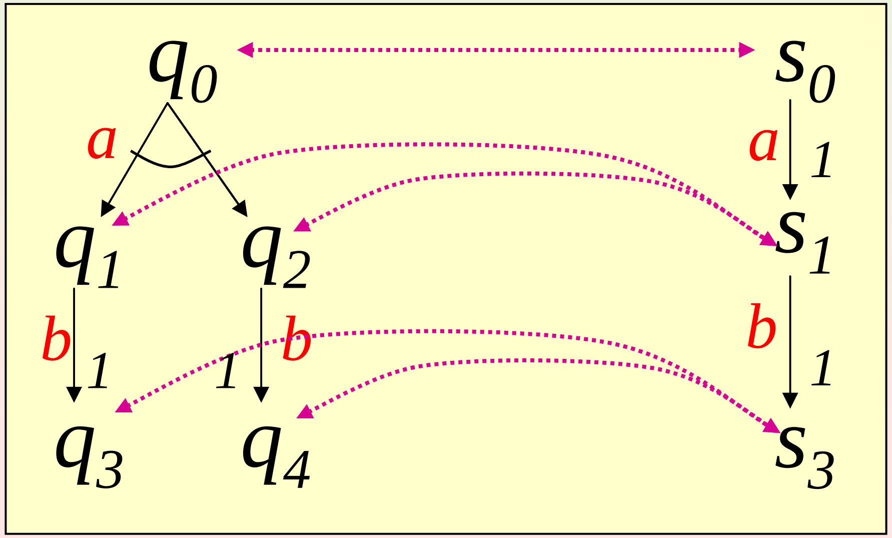
Strong bisimulation between A_1 and A_2

Relation $\boldsymbol{R} \subseteq Q \times Q$,

$Q=Q_1 \uplus Q_2$, such that

$$\forall\, q, s, a, \mu \;\exists\, \mu'$$

$$\begin{tikzcd} q \arrow[r, "a"] \arrow[d, dash, dotted, "\boldsymbol{R}"'] & \mu \arrow[d, dash, dotted, "\boldsymbol{R}"] \\ s \arrow[r, "a"] & \mu' \end{tikzcd}$$



[LS89]

$$\mu \,\boldsymbol{R}\, \mu'$$

$$\Leftrightarrow$$

$$\forall C \in Q/\boldsymbol{R} \,.\, \mu(C) = \mu'(C)$$

Weak Bisimulation on Automata

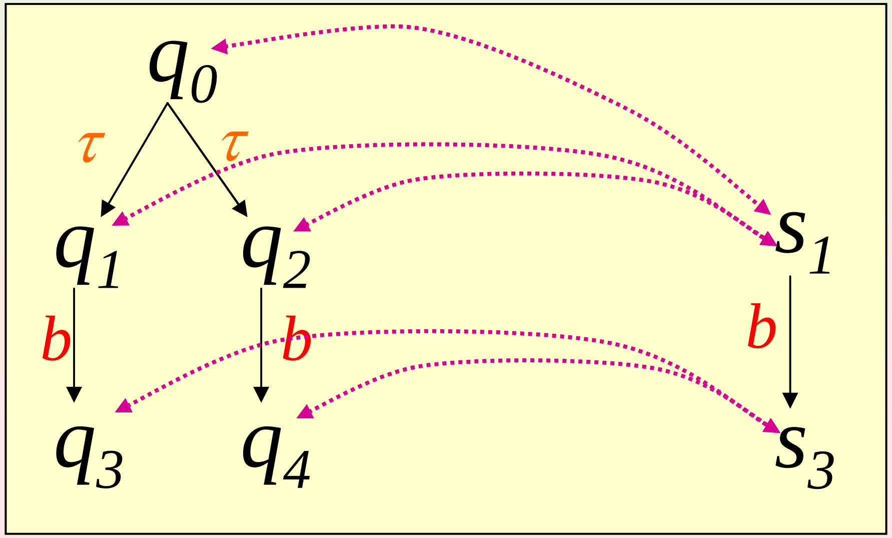
Weak bisimulation between A_1 and A_2

Relation $\boldsymbol{R} \subseteq Q \times Q$,

$Q = Q_1 \uplus Q_2$, such that

$\forall\, q, s, a, q' \;\exists\, s'$

$$\begin{tikzcd} q \arrow[r, "a"] \arrow[d, dash, dotted, "\boldsymbol{R}"'] & q' \arrow[d, dash, dotted, "\boldsymbol{R}"] \\ s \arrow[r, Rightarrow, "a"] & s' \end{tikzcd}$$



$$s \overset{a}{\Rightarrow} s'$$

$$\Leftrightarrow$$

$\exists \alpha$: $trace(\alpha) = a$, $fstate(\alpha) = s$, $lstate(\alpha) = s'$

Weak bisimulation on Probabilistic Automata

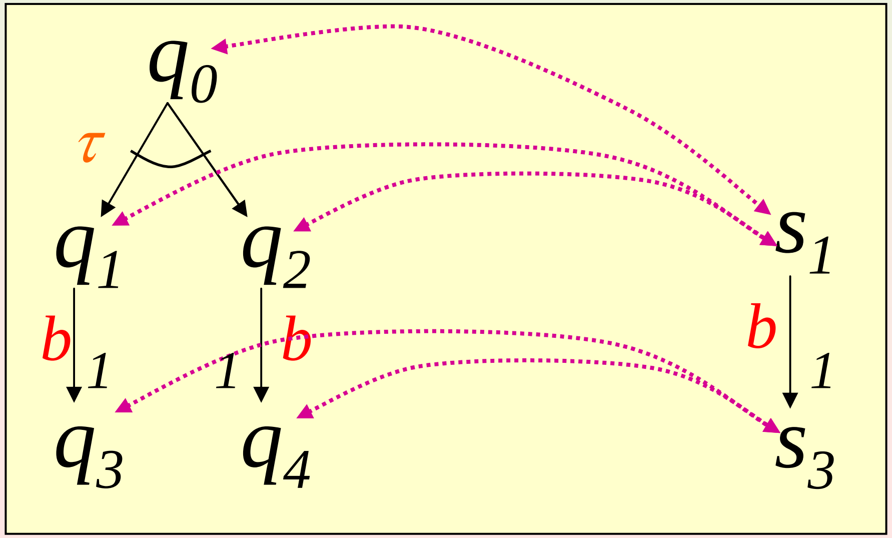
Weak bisimulation between A_1 and A_2

Relation $\boldsymbol{R} \subseteq Q \times Q$,

$Q = Q_1 \uplus Q_2$, such that

$$\forall\, q, s, a, \mu\ \exists\, \mu'$$

$$\begin{tikzcd} q \arrow[r, "a"] \arrow[d, dash, dotted, "\boldsymbol{R}"'] & \mu \arrow[d, dash, dotted, "\boldsymbol{R}"] \\ s \arrow[r, Rightarrow, "a"] & \mu' \end{tikzcd}$$



$$\mu\ \boldsymbol{R}\ \mu'$$

[LS89]

$$\Leftrightarrow$$

$$\forall C \in Q/\boldsymbol{R}\ .\ \mu(C) = \mu'(C)$$

Weak Transition

$$q \stackrel{a}{\Longrightarrow} \rho$$

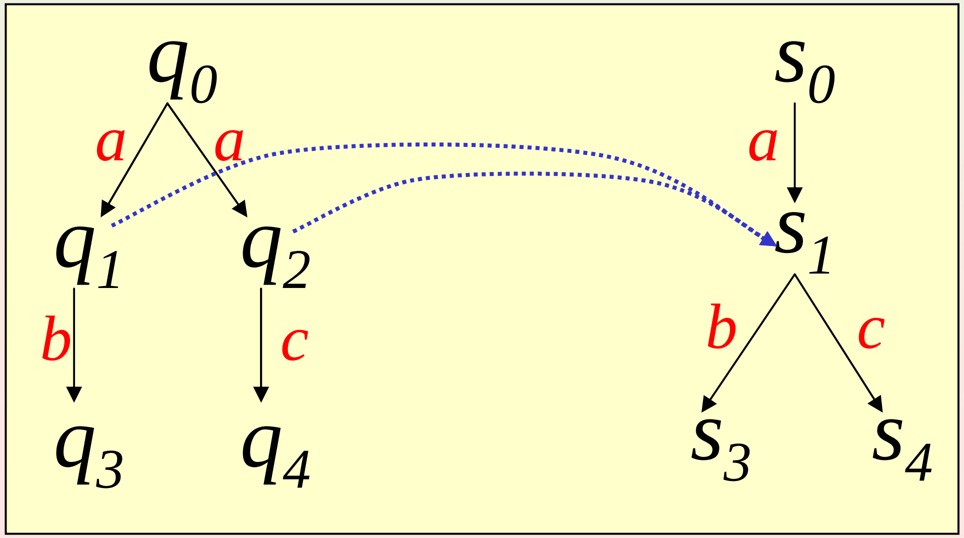
There is a probabilistic execution μ such that

- $\mu(exec^*) = 1$ (it is finite)
- $trace(\mu) = \delta(a)$ (its trace is a)
- $fstate(\mu) = \delta(q)$ (it starts from q)
- $lstate(\mu) = \rho$ (it leads to ρ)

$q \stackrel{a}{\Rightarrow} s$ iff $\exists\alpha$: $trace(\alpha)=a$, $fstate(\alpha)=q$, $lstate(\alpha)=s$

Simulations (Automata)

Forward simulation from A_1 to A_2 ($A_1 \leq_F A_2$)
Relation $R \subseteq Q_1 \times Q_2$ such that



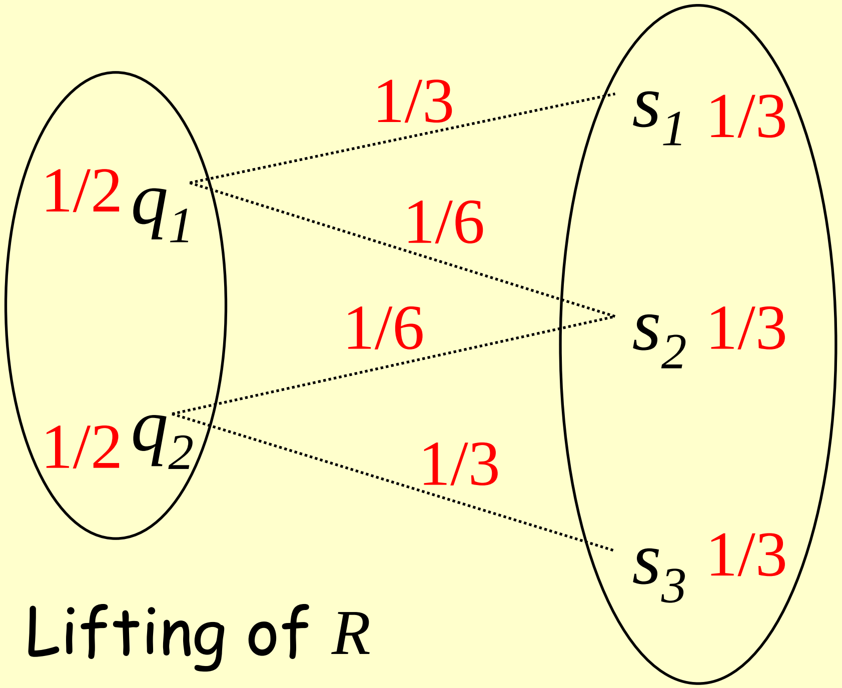
$\forall\ q, s, a, q' \ \exists\ s'$

$$
\begin{tikzcd}
s \arrow[r, "a"] \arrow[d, dash, dotted, "R"'] & s' \arrow[d, dash, dotted, "R"] \\
q \arrow[r, "a"] & q'
\end{tikzcd}
$$

Simulations on Probabilistic Automata

Simulation from A_1 to A_2 $(A_1 \leq_F A_2)$

Relation $R \subseteq Q_1 \times Q_2$ such that



Lifting of R

$\forall\, q, s, a, \mu\ \exists\, \mu'$

$$\begin{tikzcd} s \arrow[r, "a"] \arrow[d, dash, dotted, "R"'] & \mu' \arrow[d, dash, dotted, "R"] \\ q \arrow[r, "a"] & \mu \end{tikzcd}$$

… and now …

… we move to a…

Case Study

Bellare and Rogaway MAP1 Protocol

$$A \xrightarrow{R_A} B$$

$$A \xleftarrow{[B.A.R_A.R_B]_s} B$$

$$A \xrightarrow{[A.R_B]_s} B$$

- Nonces are generated randomly
- The key s is the secret for a Message Authentication Code
  - Specifically, MAC based on pseudo-random functions

Nonces

- Number ONCE
  - Typically drawn randomly

- Claim
  - For each constant c and polynomial p
  - There exists $\boldsymbol{k}$ such that for each $k \geq \boldsymbol{k}$
  - If $n_1,n_2,\ldots,n_{p(k)}$ are random nonces from $\{0,1\}^k$
  - Then $\boxed{\Pr[\exists_{i \neq j}\ n_i = n_j] < k^{-c}}$

Message Authentication Code

- Triple (G,A,V)
  - G on input 1^k generates $s \in \{0,1\}^k$
  - For each s and each a
    - Pr[V(s,a,A(s,a))=1]=1
- Forger
  - On input 1^k obtains MAC of strings of its choice
  - Outputs a pair (a,b)
  - Successful if V(s,a,b)=1 and a different from previous queries
- Secure MAC
  - Every feasible forger succeeds with negligible probability

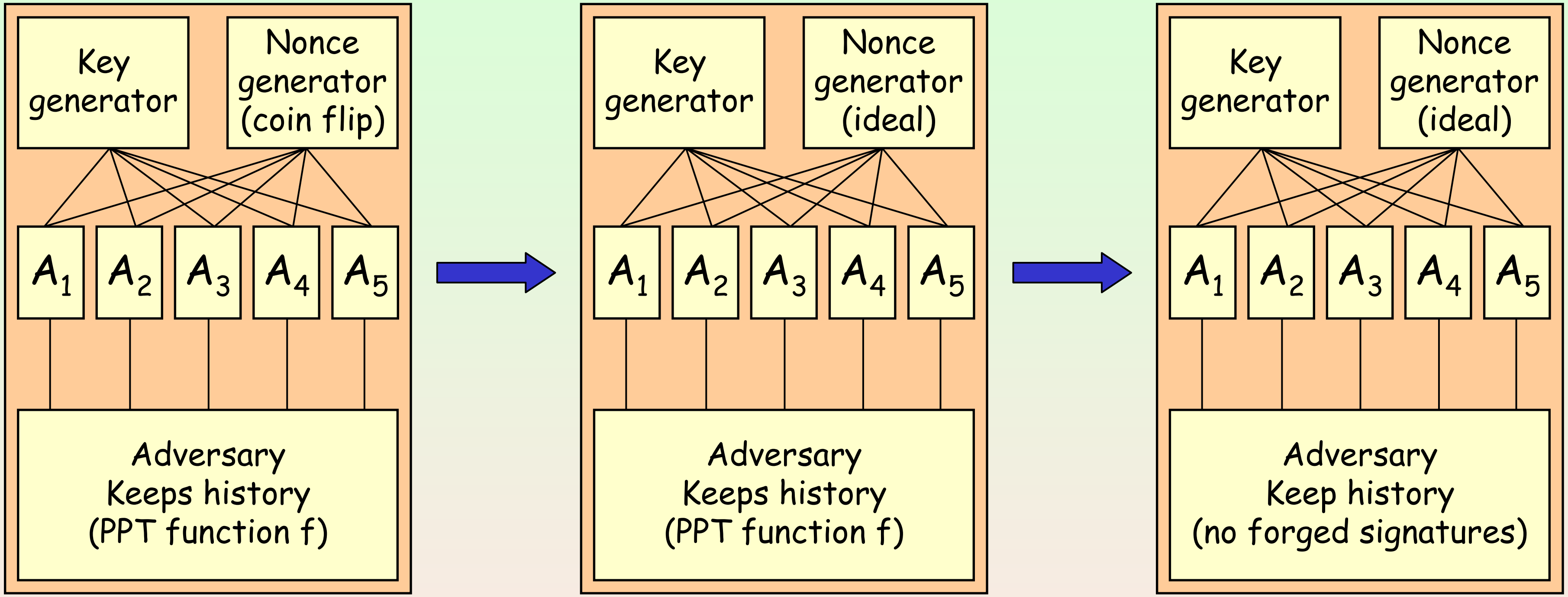
MAP1: Matching Conversations

- Matching conversation between A and B
  - Every message from A to B delivered unchanged
    - Possibly last message lost
    - Response from B returned to A
  - Every message received by A generated by B
    - Messages generated by B delivered to A
    - Possibly last message lost
- Correctness condition
  - Matching conversation implies acceptance
  - Negligible probability of acceptance without matching conversation

MAP1: Correctness Proof

- Let A be a PPT machine that interacts with the agents
- Show that A induces "no-match" with negligible probability
  - Argue that repeated nonces occur with negligible probability
  - Argue that A is an attack against a message authentication code
- Features
  - Relies on underlying pseudo-random functions
  - Proves correctness assuming truly random functions
  - Builds a distinguisher for PRFs if an attack exists
- Criticism
  - The arguments are semi-formal and not immediate
  - Three different concepts intermixed
    - Nonces
    - Message authentication codes
    - Matching conversations

MAP1: Hierarchical Analysis



- Agents indexed by X, Y, t
- Need to find suitable simulations
  - Step conditions lead to local arguments
  - Yet transitions cannot be matched exactly

Nonce Generators

- State
  - $value_{X,Y,t}$ initially $\perp$
  - $FreshNonces$ initially $\{0,1\}^k$

Coin flip

Ideal

- Transitions
  - Input $NonceRequest_{X,Y,t}$
  - Effect
    - Let $v \in_R \{0,1\}^k$
    - $value_{X,Y,t} = v$
    - $FreshNonces = FreshNonces-\{v\}$

    - Let $v \in_R FreshNonces$
  - Output $NonceResponse_{X,Y,t}(n)$
  - Precondition
    - $n = value_{X,Y,t}$
  - Effect
    - $value_{X,Y,t} = \perp$

Adversary

- Keeps a variable *history*
  - Holds all previous messages
- Real adversary
  - Runs a cycle where
    - Computes the next message to send using a PPT function f
    - Sends the message
    - Waits for the answer if expected
- Ideal adversary
  - Highly nondeterministic
  - Stores all input
  - Sends messages that do not contain forged authentications

Problems with Simulations

- Problem
  - Consider a transition of the real nonce generator
  - With some probability there is a repeated nonce
  - The ideal nonce generator does not repeat nonces
  - Thus, we cannot match the step
- Solution
  - Match transitions up to some error

Convex Combination of Measures

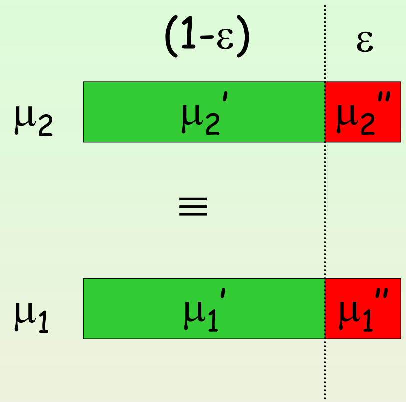
- Let μ_1 and μ_2 be probability measures
- Let p_1 and p_2 be reals in [0,1] such that $p_1+p_2=1$
- Define a new measure $\mu = p_1\mu_1+p_2\mu_2$ as follows
  - $\forall X, \mu(X) = p_1\mu_1(X)+p_2\mu_2(X)$

- Theorem: μ is a proability measure

- Same result extends to countable summation

Approximate Simulations [ST07]

- Change equivalence on measures
  - $\mu_1 \equiv_\varepsilon \mu_2$ iff
    - $\mu_1 = (1-\varepsilon)\mu_1' + \varepsilon\mu_1''$
    - $\mu_2 = (1-\varepsilon)\mu_2' + \varepsilon\mu_2''$
    - $\mu_1' \equiv \mu_2'$



- Add parameterizations
  - Consider families of PIOA parameterized by k
- Require ε smaller than any polynomial in k
  - …provided that computations are of polynomial length

Example: Approximated Lifting

$\{2/3\ q_1, 1/3\ q_2\}$ = $2/3\ \{1/2\ q_1, 1/2\ q_2\} + 1/3\{1\ q_1\}$

? $\varepsilon = 1/3$

$\{1/3\ s_1, 1/3\ s_2, 1/3\ s_3\}$ = $2/3\ \{1/2\ s_1, 1/2\ s_2\} + 1/3\{1\ s_3\}$

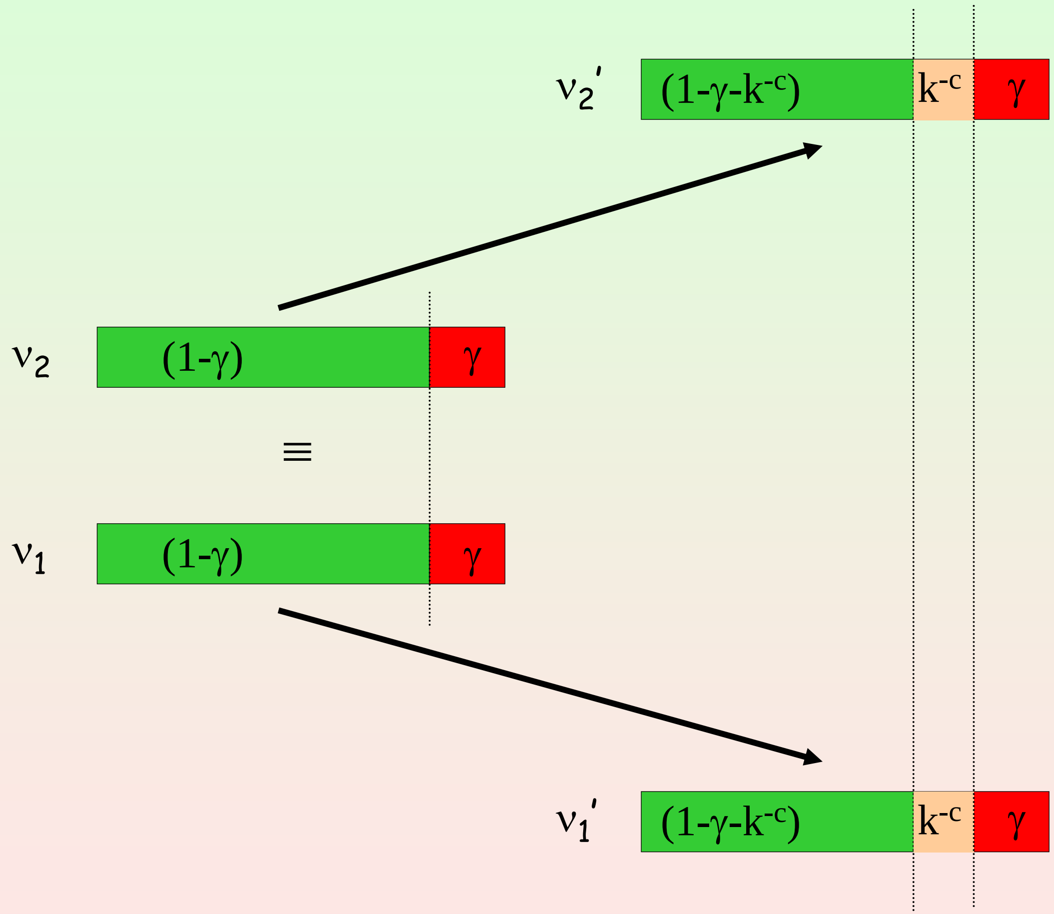
Approximate Simulations

$$\{A_k\} \ \{R_k\} \ \{B_k\}$$

- For each constant c and polynomial p
- There exists $\boldsymbol{k}$ such that for each $k \geq \boldsymbol{k}$
- Whenever
  - ν_1 reached within $p(k)$ steps in A_k
  - $\nu_1 \; L(R_k,\gamma) \; \nu_2$
  - $\nu_1 \rightarrow \nu_1'$
- There exists ν_2' such that
  - $\nu_2 \rightarrow \nu_2'$
  - $\nu_1' \; L(R_k,\gamma+k^{-c}) \; \nu_2'$

$$\begin{tikzcd}
\nu_2 \arrow[r] \arrow[d, dash, dotted, "\gamma"'] & \nu_2' \arrow[d, dash, dotted, "\gamma+k^{-c}"] \\
\nu_1 \arrow[r] & \nu_1'
\end{tikzcd}$$

Approximate Simulations
Step Condition



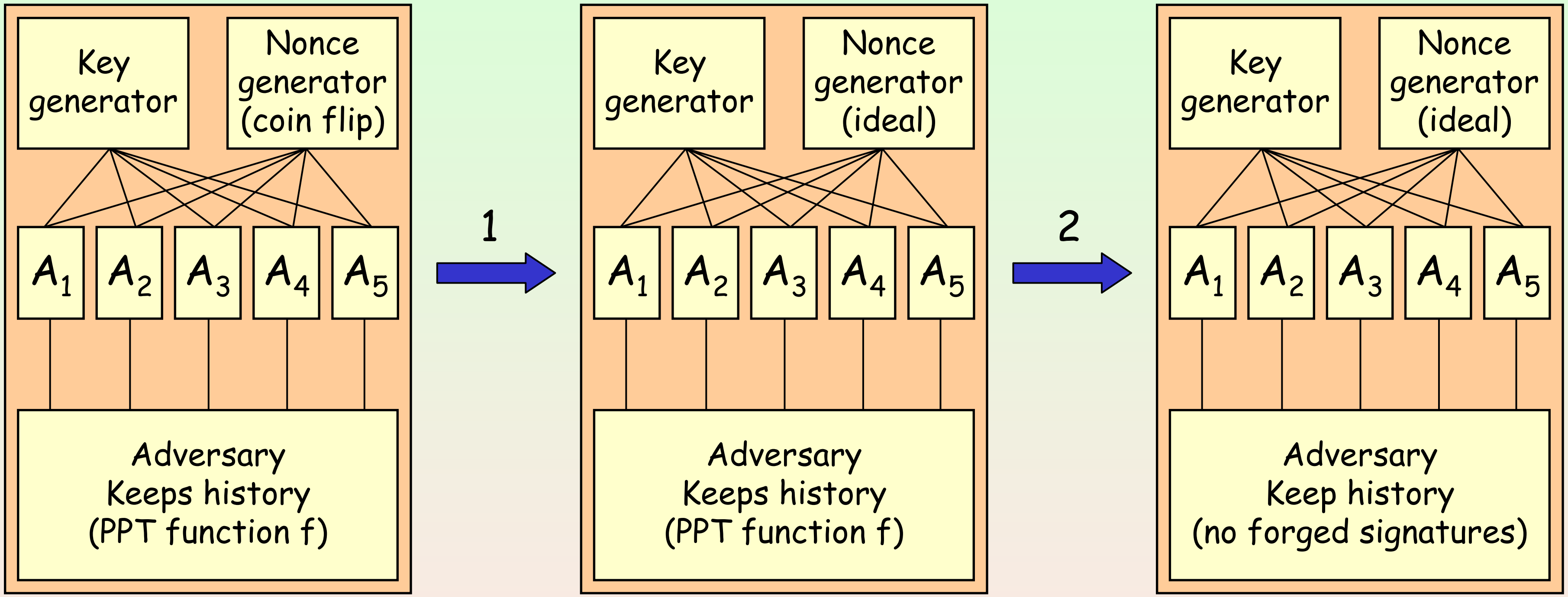
Execution Correspondence under Approximated Simulations

If $\{A_k\}$ $\{R_k\}$ $\{B_k\}$ then

- For each constant c and polynomial p
- There exists $\boldsymbol{k}$ such that for each $k \geq \boldsymbol{k}$
- For each scheduler σ_1
  - ν_1 reached within $p(k)$ steps in A_k with σ_1
- There exists σ_2 such that
  - ν_2 reached within $p(k)$ steps in B_k with σ_2
  - $\nu_1 \; L(R_k, p(k)k^{-c}) \; \nu_2$

- Observation
  - $p(k)k^{-c}$ can be smaller than any $k^{-c'}$ by choosing $c=c'+degree(p)$

Example: Approximate Simulations Bellare-Rogaway MAP1 Protocol

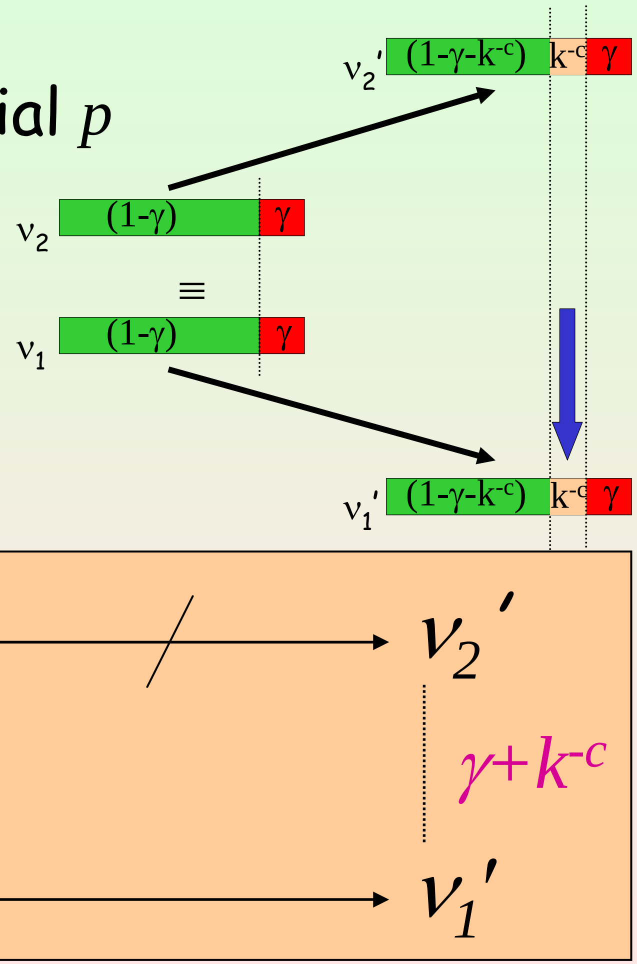


- Negation of the step condition
  - 1: Two random nonces are equal with high probability
  - 2: Function f defines a forger for a signature scheme

Negation of Step Condition

$\{A_k\}\ \{R_k\}\ \{B_k\}$

- There exists constant c and polynomial p
- For each $\boldsymbol{k}$ there exists $k \geq \boldsymbol{k}$
- There exists
  - ν_1 reached within $p(k)$ steps in A_k
  - $\nu_1\ L(R_k,\gamma)\ \nu_2$
  - $\nu_1 \rightarrow \nu_1'$
- There is no ν_2' such that
  - $\nu_2 \rightarrow \nu_2'$
  - $\nu_1'\ L(R_k,\gamma+k^{-c})\ \nu_2'$



- Signature forged in v_1'
  - Probability at least k^{-c}

Nonces

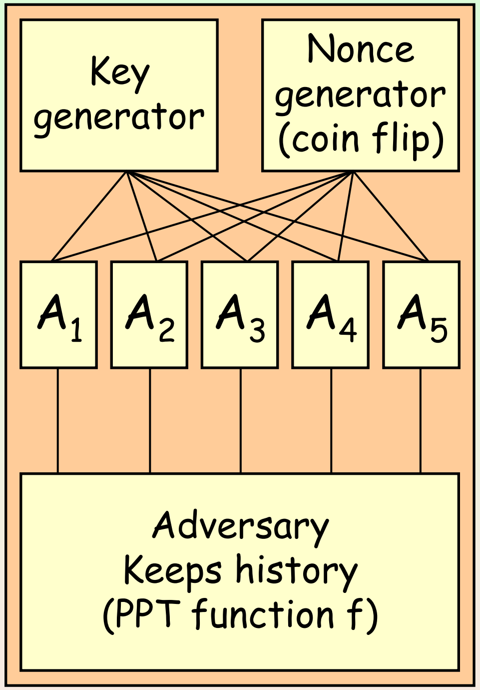
- Number ONCE
  - Typically drawn randomly

- Claim
  - For each constant c and polynomial p
  - There exists $\boldsymbol{k}$ such that for each $k \geq \boldsymbol{k}$
  - If $n_1, n_2, \ldots, n_{p(k)}$ are random nonces from $\{0,1\}^k$
  - Then $\boxed{\Pr[\exists_{i \neq j}\, n_i = n_j] < k^{-c}}$

Applicability

- Dolev-Yao Model
  - Soundness w.r.t. indistinguishability
  - How about correspondence of computations?
- Cryptographic library
  - More rigorous/local proofs?
  - Alternative to error sets?
- Game transformations
  - Proof method?

Problems with Nondeterminism MAP1 Protocol [BR93]



- Authentication protocol
  - Symmetric key signature schema
  - Computational Dolev-Yao
  - Adversary queries agents
- Potential problems
  - Let s be the shared key
  - Adversary queries k agents
  - Agent i replies if i^{th} bit of s is 1
  - The adversary knows the shared key
- Solution
  - One query at a time
  - Wait for the answer (agents as oracles)

Current Status

- What we have
  - A notion of task PIOA with restricted schedulers
    - Task: equivalence relation on actions
    - Equivalence relation on states
      - Preserve task enabledness
      - Each state enables at most one action for each task
      - Each transition reaches only one task
  - A notion of approximated language inclusion
    - For each trace distribution of A there exists an indistinguishable trace distribution of B
  - A notion of exact simulation safe for language inclusion
    - Works on task PIOAs
  - A notion of aproximated simulation
    - Works for PAs

Current Status

- … what we have
  - Analysis of oblivious transfer in UC framework
    - Task PIOAs as model
    - Hierarchical verification via simulations
    - Crypto-steps via approximated language inclusion
  - Analysis of MAP1 protocol
    - PAs as model
    - Approximated simulations as technique
    - Mixture of Dolev-Yao and computational
    - No restriction of nondeterminism
      - Yet accurate description of objects

Current Status

- What we do not have
  - Connections
    - Approximated simulations with
      - Approximated language inclusion
      - Restricted schedulers
    - Semantics
      - Metrics and exact equivalences
  - Flexibility on restrictions
    - Task PIOAs are very restrictive
      - … though they work
      - Chatzikokolakis and Palamidessi may help (Concur07)
  - Understanding of restrictions
    - Are we restricting too much?

What Else?

- A lot to understand on approximated simulations
  - Are they connected to metrics?
  - Can we define them incrementally
    - How far can we go without polynomial bounds?
  - How about approximated language inclusion?

- Need more techniques
  - Can we have a uniform view?
  - Can we relate better computational and symbolic approaches?
  - Any crucial differences between crypto-primitives and protocols?
  - How about cross migration of techniques?

- Need more automation
  - … but we need to understand what we automate