# Interactive Proofs

Lecture 18
AM

# Interactive Proofs

# Interactive Proofs

- IP[k]

# Interactive Proofs

- IP[k]
  - IP[poly] = PSPACE

# Interactive Proofs

- IP[k]
  - IP[poly] = PSPACE
    - IP protocol for TQBF using arithmetization

# Interactive Proofs

- IP[k]
  - IP[poly] = PSPACE
    - IP protocol for TQBF using arithmetization
      - We saw IP protocol for sum-check

# Interactive Proofs

- IP[k]
  - IP[poly] = PSPACE
    - IP protocol for TQBF using arithmetization
      - We saw IP protocol for sum-check
  - IP[const] = AM[const]

# Interactive Proofs

- IP[k]
  - IP[poly] = PSPACE
    - IP protocol for TQBF using arithmetization
      - We saw IP protocol for sum-check
  - IP[const] = AM[const]
    - We saw public coin protocol for Graph Non-Isomorphism

# Interactive Proofs

- IP[k]
  - IP[poly] = PSPACE
    - IP protocol for TQBF using arithmetization
      - We saw IP protocol for sum-check
  - IP[const] = AM[const]
    - We saw public coin protocol for Graph Non-Isomorphism
      - Using 2-universal hash functions

# Interactive Proofs

- IP[k]
  - IP[poly] = PSPACE
    - IP protocol for TQBF using arithmetization
      - We saw IP protocol for sum-check
  - IP[const] = AM[const]
    - We saw public coin protocol for Graph Non-Isomorphism
      - Using 2-universal hash functions
- Today: Collapse of the AM hierarchy

# Interactive Proofs

- IP[k]
  - IP[poly] = PSPACE
    - IP protocol for TQBF using arithmetization
      - We saw IP protocol for sum-check
  - IP[const] = AM[const]
    - We saw public coin protocol for Graph Non-Isomorphism
      - Using 2-universal hash functions
- Today: Collapse of the AM hierarchy
  - AM[const] = AM[2]

# Recall AM

# Recall AM

# Recall AM

- AM[2] (or simply AM)

# Recall AM

- AM[2] (or simply AM)
  - Input x

# Recall AM

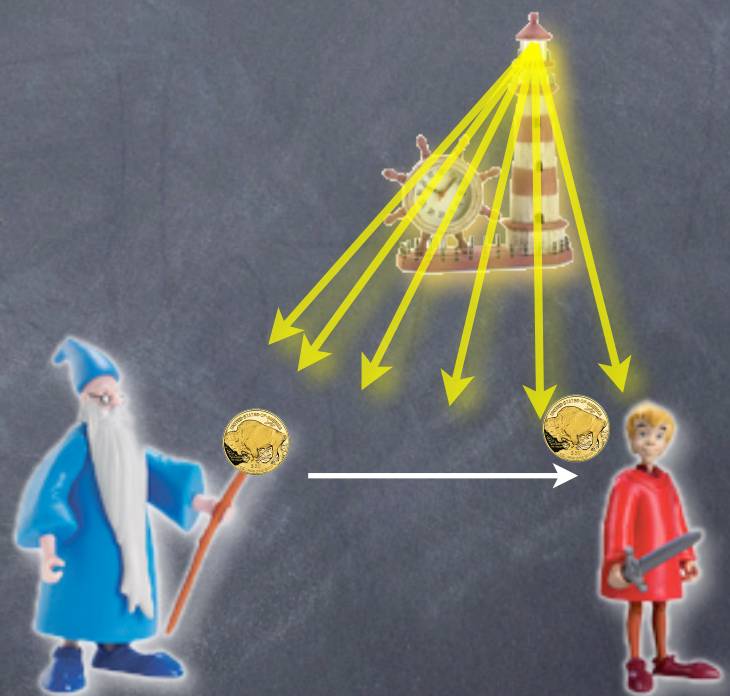
- AM[2] (or simply AM)
  - Input x
  - Random coins r come from a beacon

# Recall AM

- AM[2] (or simply AM)
  - Input x
  - Random coins r come from a beacon

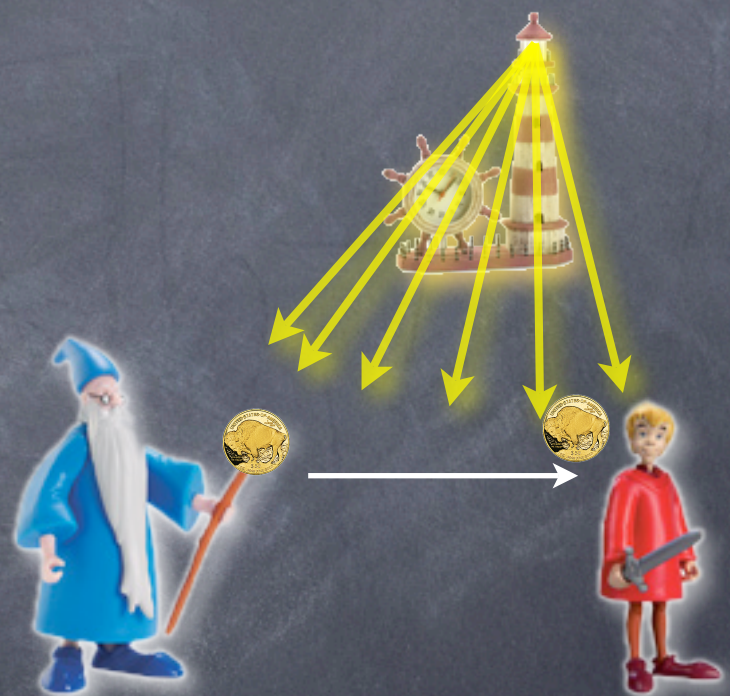
# Recall AM

- AM[2] (or simply AM)
  - Input x
  - Random coins r come from a beacon
  - Unbounded prover Merlin sends a "proof" message a



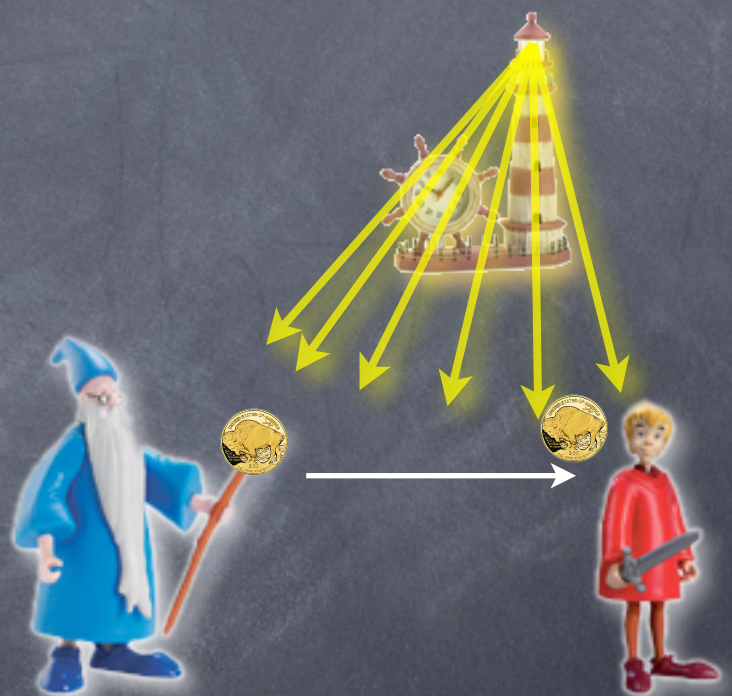
# Recall AM

- AM[2] (or simply AM)
  - Input x
  - Random coins r come from a beacon
  - Unbounded prover Merlin sends a "proof" message a
  - Polynomial time verifier Arthur runs a deterministic verification procedure R(x;r,a), and outputs Yes or No



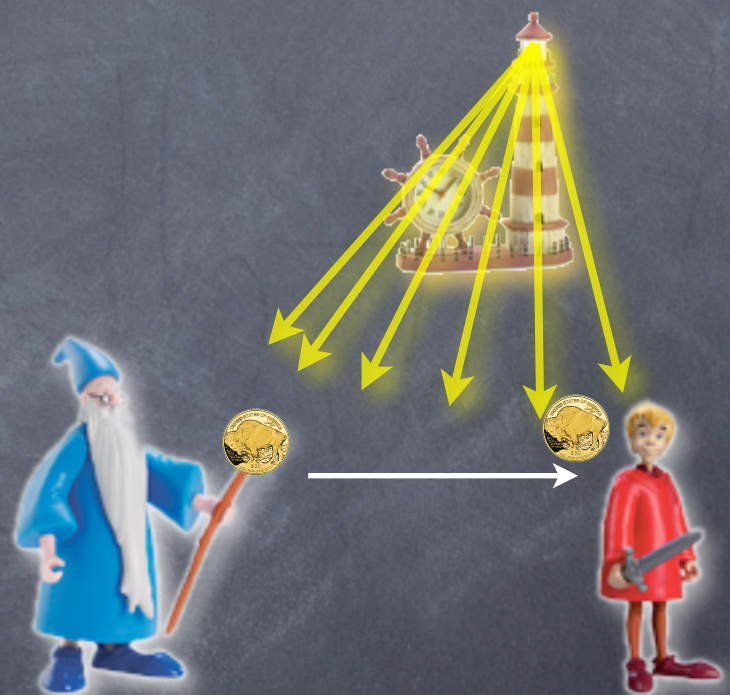
# Recall AM

- AM[2] (or simply AM)
  - Input x
  - Random coins r come from a beacon
  - Unbounded prover Merlin sends a "proof" message a
  - Polynomial time verifier Arthur runs a deterministic verification procedure R(x;r,a), and outputs Yes or No
- L is said to have an AM protocol if



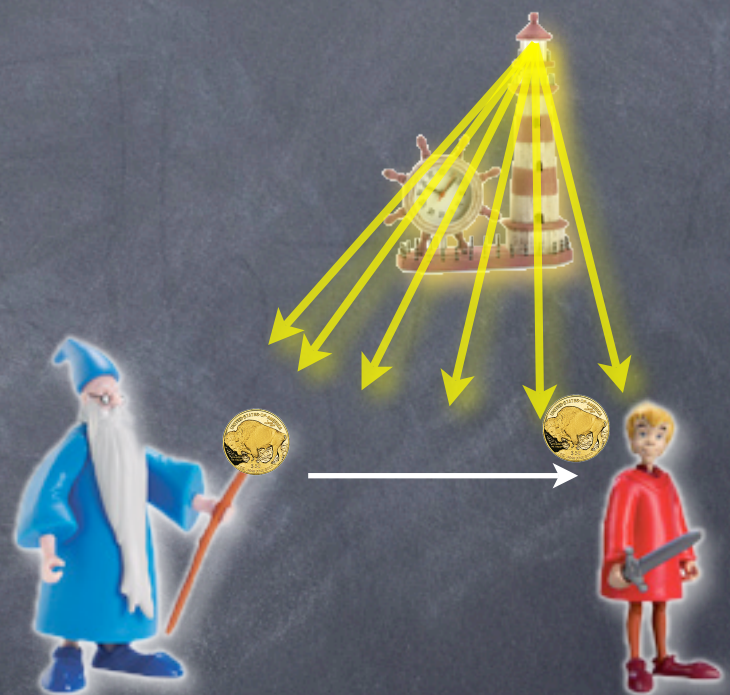
# Recall AM

- AM[2] (or simply AM)
  - Input x
  - Random coins r come from a beacon
  - Unbounded prover Merlin sends a "proof" message a
  - Polynomial time verifier Arthur runs a deterministic verification procedure R(x;r,a), and outputs Yes or No
- L is said to have an AM protocol if
  - $x \in L \Leftrightarrow \max \Pr[\text{Yes}] > 2/3$



# Recall AM

- AM[2] (or simply AM)
  - Input x
  - Random coins r come from a beacon
  - Unbounded prover Merlin sends a "proof" message a
  - Polynomial time verifier Arthur runs a deterministic verification procedure R(x;r,a), and outputs Yes or No
- L is said to have an AM protocol if
  - $x \in L \Leftrightarrow \max \Pr[\text{Yes}] > 2/3$
  - $x \notin L \Leftrightarrow \max \Pr[\text{Yes}] < 1/3$



# max Pr[Yes]

# max Pr[Yes]

- Quantity of interest

# max Pr[Yes]

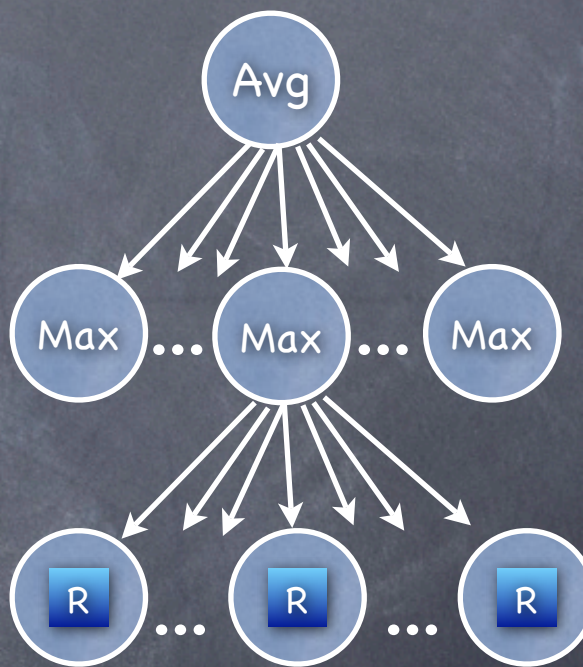
- Quantity of interest
  - Maximum (over prover strategies) probability (over coins from the beacon) of Arthur saying yes

# max Pr[Yes]

- Quantity of interest
  - Maximum (over prover strategies) probability (over coins from the beacon) of Arthur saying yes
  - Evaluate the "Avg-Max tree"

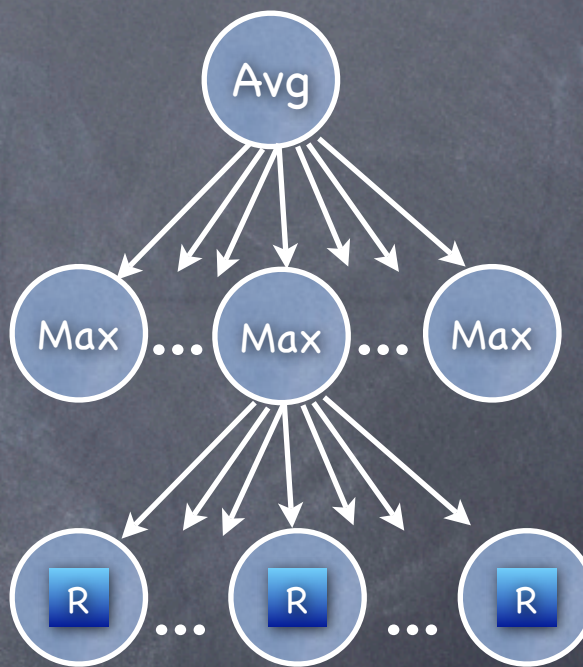
# max Pr[Yes]

- Quantity of interest
  - Maximum (over prover strategies) probability (over coins from the beacon) of Arthur saying yes
  - Evaluate the "Avg-Max tree"



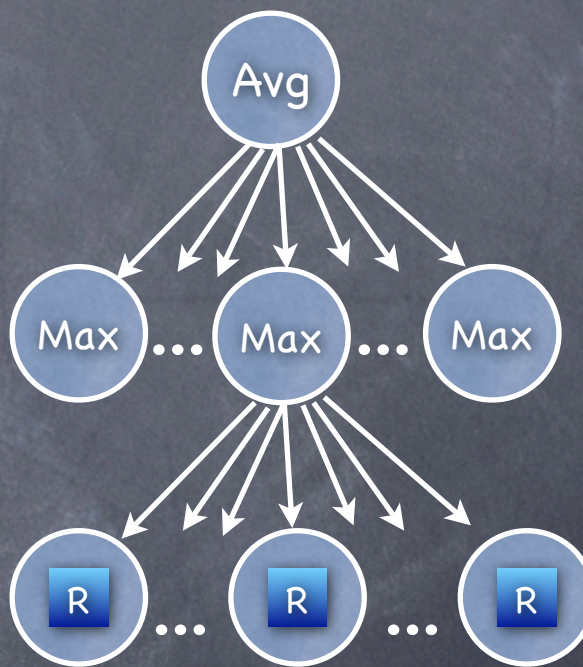
# max Pr[Yes]

- Quantity of interest
  - Maximum (over prover strategies) probability (over coins from the beacon) of Arthur saying yes
  - Evaluate the "Avg-Max tree"
    - Leaves: Pr[yes] = 0 or 1, as determined by Arthur's program



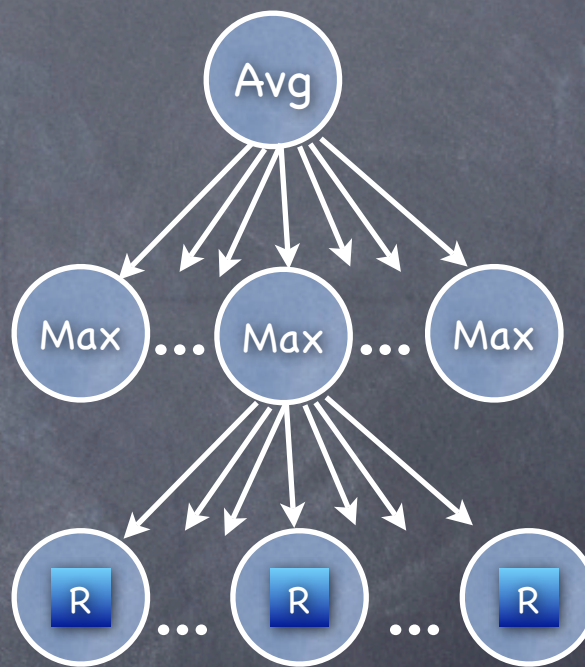
# max Pr[Yes]

- Quantity of interest
  - Maximum (over prover strategies) probability (over coins from the beacon) of Arthur saying yes
  - Evaluate the "Avg-Max tree"
    - Leaves: Pr[yes] = 0 or 1, as determined by Arthur's program
    - Max nodes: maximum of children



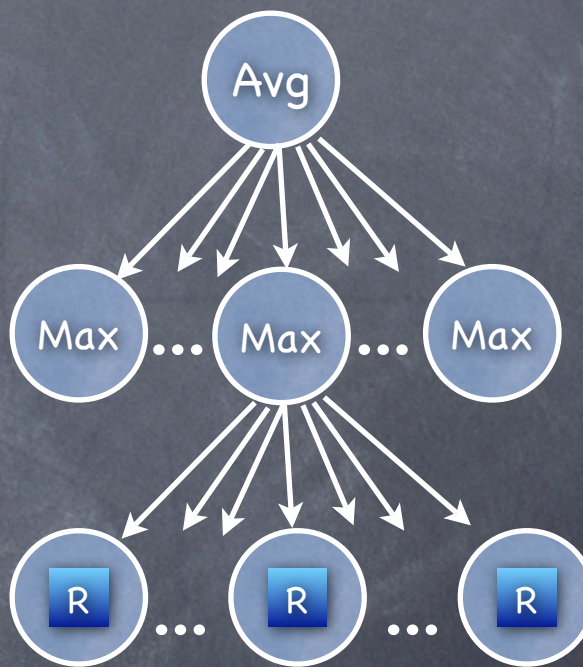
# max Pr[Yes]

- Quantity of interest
  - Maximum (over prover strategies) probability (over coins from the beacon) of Arthur saying yes
  - Evaluate the "Avg-Max tree"
    - Leaves: Pr[yes] = 0 or 1, as determined by Arthur's program
    - Max nodes: maximum of children
    - Avg node: average of children



# max Pr[Yes]

- Quantity of interest
  - Maximum (over prover strategies) probability (over coins from the beacon) of Arthur saying yes
  - Evaluate the "Avg-Max tree"
    - Leaves: Pr[yes] = 0 or 1, as determined by Arthur's program
    - Max nodes: maximum of children
    - Avg node: average of children
  - Extends to AM[k], with k alternating levels



# Soundness Amplification

# Soundness Amplification

- Recall error reduction in BPP algorithms

# Soundness Amplification

- Recall error reduction in BPP algorithms
  - By repeating and taking majority

# Soundness Amplification

- Recall error reduction in BPP algorithms
  - By repeating and taking majority
  - Exponential error reduction (by Chernoff bound)

# Soundness Amplification

- Recall error reduction in BPP algorithms
  - By repeating and taking majority
  - Exponential error reduction (by Chernoff bound)
- Extends to MA

# Soundness Amplification

- Recall error reduction in BPP algorithms
  - By repeating and taking majority
  - Exponential error reduction (by Chernoff bound)
- Extends to MA
  - Given input and any answer from Merlin, to determine Pr[Yes]

# Soundness Amplification

- Recall error reduction in BPP algorithms
  - By repeating and taking majority
  - Exponential error reduction (by Chernoff bound)
- Extends to MA
  - Given input and any answer from Merlin, to determine Pr[Yes]
  - Run many independent verifications (using independent random strings from the beacon). Chernoff bound holds.

# Soundness Amplification

- Recall error reduction in BPP algorithms
  - By repeating and taking majority
  - Exponential error reduction (by Chernoff bound)
- Extends to MA
  - Given input and any answer from Merlin, to determine Pr[Yes]
  - Run many independent verifications (using independent random strings from the beacon). Chernoff bound holds.
    - Increased the length of the second message

# Parallel Repetition for AM[k]

# Parallel Repetition for AM[k]

- Soundness amplification by sequential repetition/majority

# Parallel Repetition for AM[k]

- Soundness amplification by sequential repetition/majority
  - Exponential amplification, just like in MA. But be careful! Not independent executions (Merlin can adapt strategy over the repetitions.) But not a problem!

# Parallel Repetition for AM[k]

- Soundness amplification by sequential repetition/majority
  - Exponential amplification, just like in MA. But be careful! Not independent executions (Merlin can adapt strategy over the repetitions.) But not a problem!
  - But increases rounds

# Parallel Repetition for AM[k]

- Soundness amplification by sequential repetition/majority
  - Exponential amplification, just like in MA. But be careful! Not independent executions (Merlin can adapt strategy over the repetitions.) But not a problem!
  - But increases rounds
- Soundness amplification without increasing rounds

# Parallel Repetition for AM[k]

- Soundness amplification by sequential repetition/majority
  - Exponential amplification, just like in MA. But be careful! Not independent executions (Merlin can adapt strategy over the repetitions.) But not a problem!
  - But increases rounds
- Soundness amplification without increasing rounds
  - Parallel repetition

# Parallel Repetition for AM[k]

- Soundness amplification by sequential repetition/majority
  - Exponential amplification, just like in MA. But be careful! Not independent executions (Merlin can adapt strategy over the repetitions.) But not a problem!
  - But increases rounds
- Soundness amplification without increasing rounds
  - Parallel repetition
  - More careful! Merlin's answers (and probability of proof being rejected) in the parallel sessions could be correlated

# Parallel Repetition for AM[k]

- Soundness amplification by sequential repetition/majority
  - Exponential amplification, just like in MA. But be careful! Not independent executions (Merlin can adapt strategy over the repetitions.) But not a problem!
  - But increases rounds
- Soundness amplification without increasing rounds
  - Parallel repetition
  - More careful! Merlin's answers (and probability of proof being rejected) in the parallel sessions could be correlated
  - Still turns out to give exponential amplification

# MA ⊆ AM

# MA ⊆ AM

- Publishing random test before receiving proof

# MA ⊆ AM

- Publishing random test before receiving proof
  - Completeness is no worse

# MA ⊆ AM

- Publishing random test before receiving proof
  - Completeness is no worse
  - If MA soundness error was sufficiently small, can use union bound over all Merlin messages to get that the AM soundness error is still small

# MA ⊆ AM

- Publishing random test before receiving proof
  - Completeness is no worse
  - If MA soundness error was sufficiently small, can use union bound over all Merlin messages to get that the AM soundness error is still small
    - If MA soundness error < $1/2^{m+2}$, where m is the length of Merlin's message, AM soundness error < 1/4

# MA ⊆ AM

- Publishing random test before receiving proof
  - Completeness is no worse
  - If MA soundness error was sufficiently small, can use union bound over all Merlin messages to get that the AM soundness error is still small
    - If MA soundness error < $1/2^{m+2}$, where m is the length of Merlin's message, AM soundness error < 1/4
- Note: Argument similar to why BPP ⊆ P/poly

# MA ⊆ AM

- Publishing random test before receiving proof
  - Completeness is no worse
  - If MA soundness error was sufficiently small, can use union bound over all Merlin messages to get that the AM soundness error is still small
    - If MA soundness error < $1/2^{m+2}$, where m is the length of Merlin's message, AM soundness error < 1/4
- Note: Argument similar to why BPP ⊆ P/poly
- Extends to MAM ⊆ AM

# MA ⊆ AM

- Publishing random test before receiving proof
  - Completeness is no worse
  - If MA soundness error was sufficiently small, can use union bound over all Merlin messages to get that the AM soundness error is still small
    - If MA soundness error < $1/2^{m+2}$, where m is the length of Merlin's message, AM soundness error < 1/4
- Note: Argument similar to why BPP ⊆ P/poly
- Extends to MAM ⊆ AM
  - So MAM = AM

# Collapse of the AM hierarchy

# Collapse of the AM hierarchy

- Intuition: Can change any MA sequence to an AM sequence

# Collapse of the AM hierarchy

- Intuition: Can change any MA sequence to an AM sequence
  - Need a notion of soundness error in each round

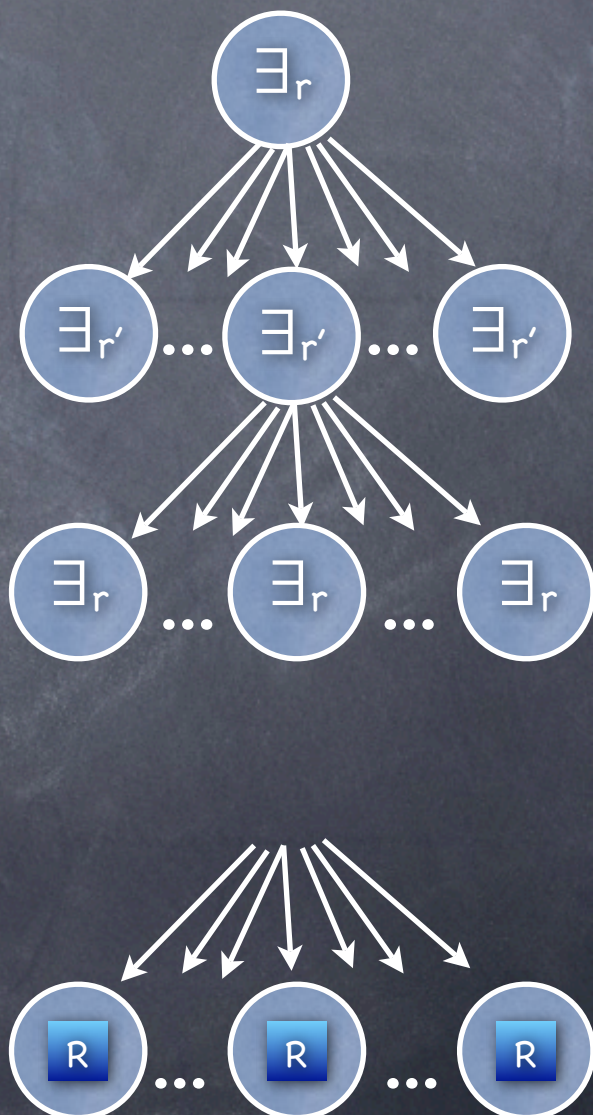
# Alternating Threshold TM

# Alternating Threshold TM

- A generalization of ATM, with two thresholds instead of $\exists$ and $\forall$

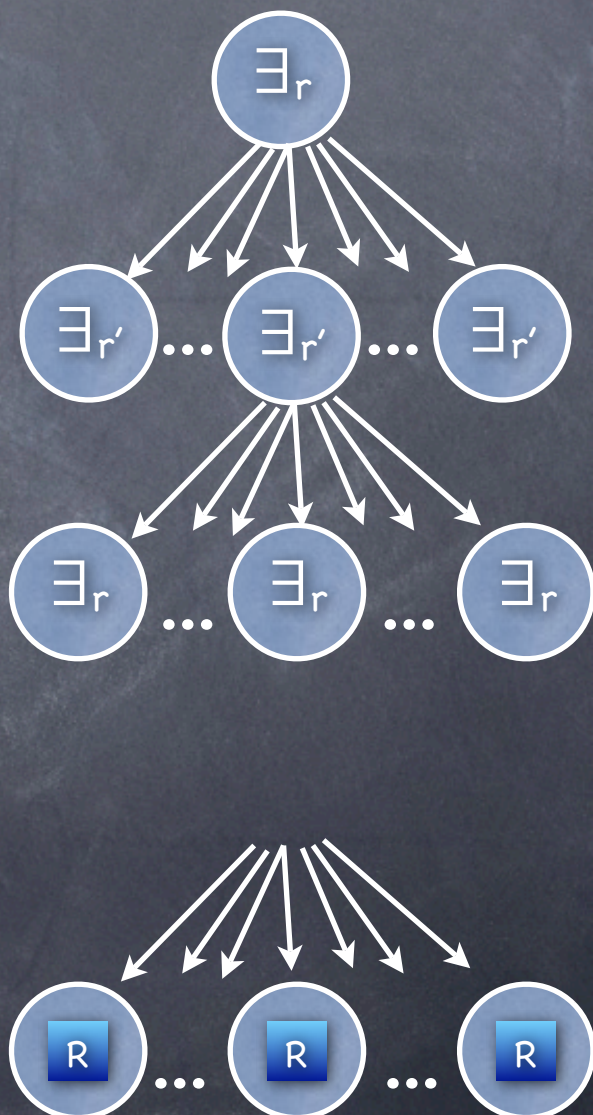
# Alternating Threshold TM

- A generalization of ATM, with two thresholds instead of ∃ and ∀



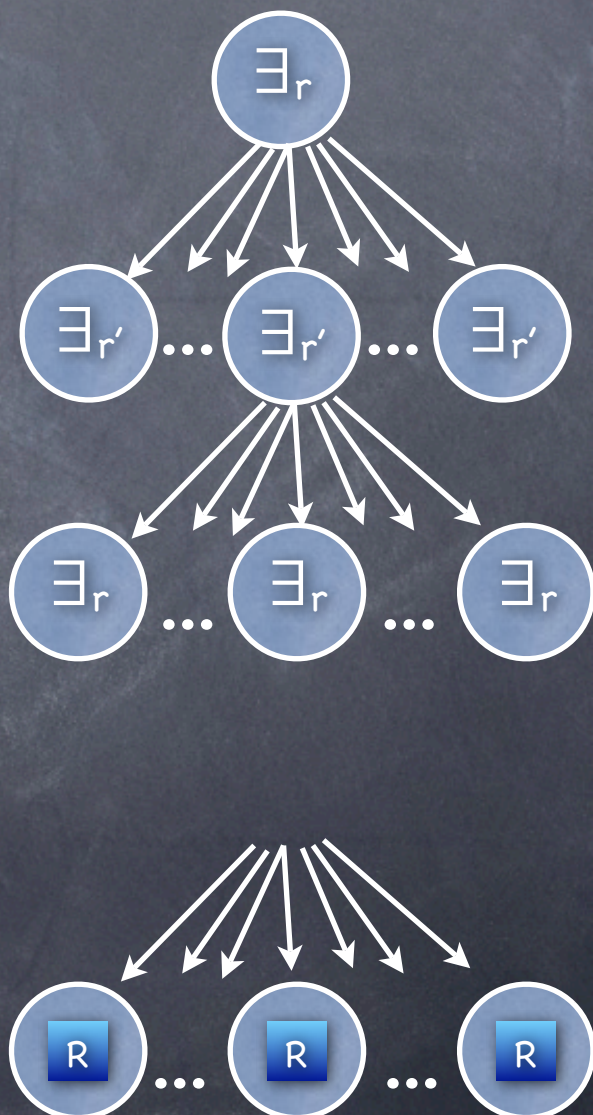
# Alternating Threshold TM

- A generalization of ATM, with two thresholds instead of ∃ and ∀
  - $\exists_r$: ≥ (or >) r fraction of children are 1?



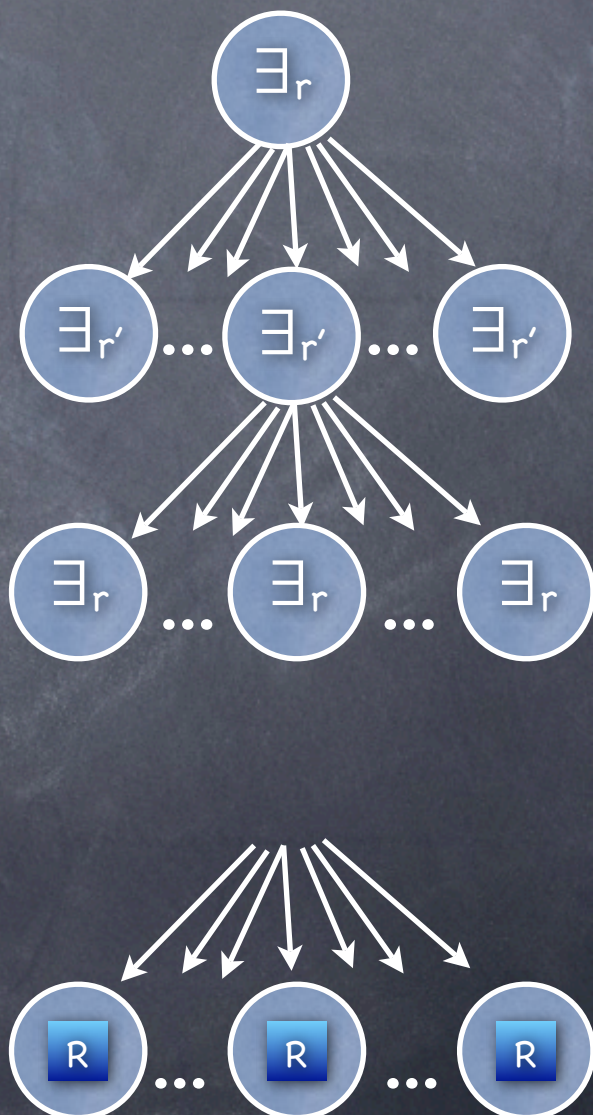
# Alternating Threshold TM

- A generalization of ATM, with two thresholds instead of ∃ and ∀
  - $\exists_r$: ≥ (or >) r fraction of children are 1?
    - $\exists_0$ is ∃, and $\exists_1$ is ∀



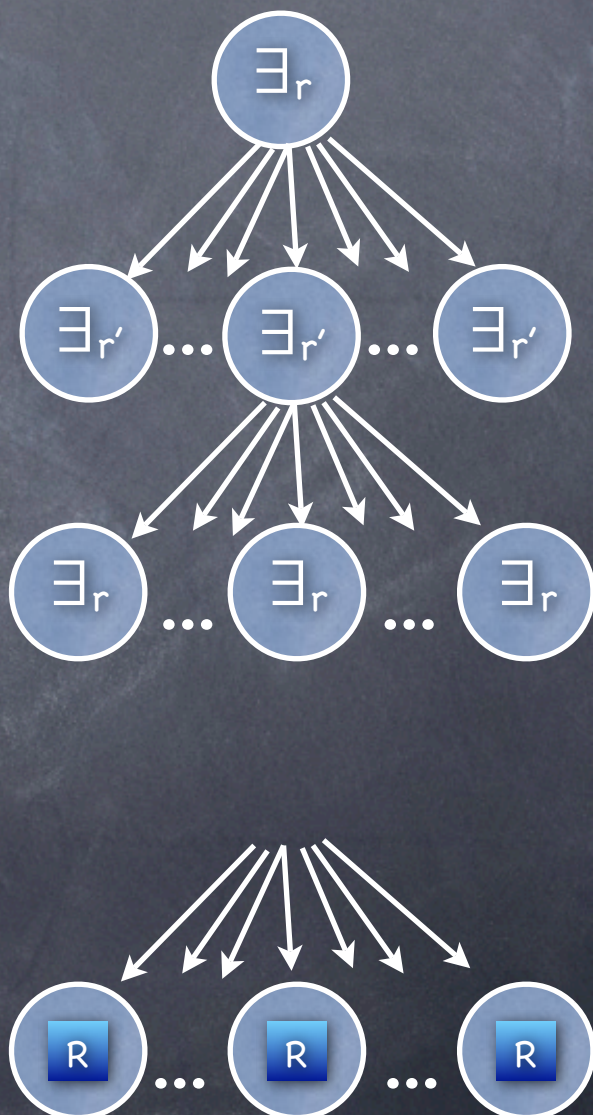
# Alternating Threshold TM

- A generalization of ATM, with two thresholds instead of ∃ and ∀
  - $\exists_r$: ≥ (or >) r fraction of children are 1?
    - $\exists_0$ is ∃, and $\exists_1$ is ∀
  - Leaves R(x;path) = 0 or 1



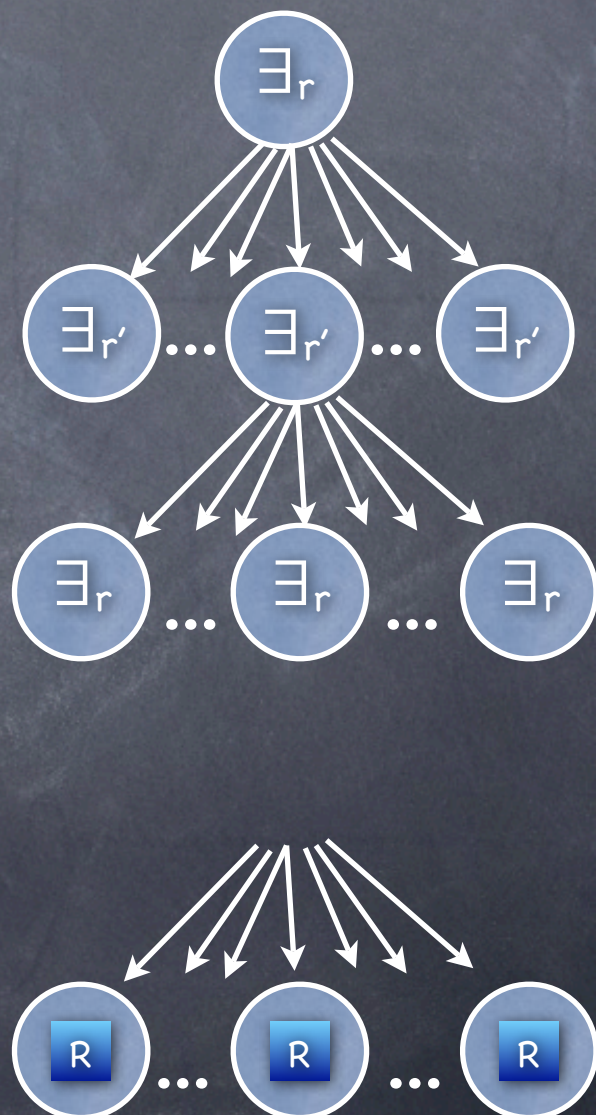
# Alternating Threshold TM

- A generalization of ATM, with two thresholds instead of ∃ and ∀
  - $\exists_r$: ≥ (or >) r fraction of children are 1?
    - $\exists_0$ is ∃, and $\exists_1$ is ∀
  - Leaves R(x;path) = 0 or 1
- Parameters: depth (number of alternations) and size = log(#leaves) (= total length of the "messages")



# Alternating Threshold TM

- A generalization of ATM, with two thresholds instead of ∃ and ∀
  - $\exists_r$: ≥ (or >) r fraction of children are 1?
    - $\exists_0$ is ∃, and $\exists_1$ is ∀
  - Leaves R(x;path) = 0 or 1
- Parameters: depth (number of alternations) and size = log(#leaves) (= total length of the "messages")
  - Will denote as ATTM[k,(r,r'),R] (size and individual degrees implicit)



# Alternating Threshold TM

# Alternating Threshold TM

- We will be interested in ATTM[k,(r,r'),R] where

# Alternating Threshold TM

- We will be interested in ATTM[k,(r,r'),R] where
  - One of r, r' is a fraction > 1/2 (called the threshold), and the other is 0 or 1

# Alternating Threshold TM

- We will be interested in ATTM[k,(r,r'),R] where
  - One of r, r' is a fraction > 1/2 (called the threshold), and the other is 0 or 1
  - k is constant, size is polynomial and R is a polynomial time relation

# Alternating Threshold TM

- We will be interested in ATTM[k,(r,r'),R] where
  - One of r, r' is a fraction > 1/2 (called the threshold), and the other is 0 or 1
  - k is constant, size is polynomial and R is a polynomial time relation
- ATTM threshold can also be amplified using "parallel repetition"!

# Alternating Threshold TM

- We will be interested in ATTM[k,(r,r'),R] where
  - One of r, r' is a fraction > 1/2 (called the threshold), and the other is 0 or 1
  - k is constant, size is polynomial and R is a polynomial time relation
- ATTM threshold can also be amplified using "parallel repetition"!
  - Takes threshold from (1/2 + c) to $(1 - 1/2^n)$

# Alternating Threshold TM

- We will be interested in ATTM[k,(r,r'),R] where
  - One of r, r' is a fraction > 1/2 (called the threshold), and the other is 0 or 1
  - k is constant, size is polynomial and R is a polynomial time relation
- ATTM threshold can also be amplified using "parallel repetition"!
  - Takes threshold from (1/2 + c) to $(1 - 1/2^n)$
  - k unchanged, size increases by a polynomial factor

# A Pair of Complementary ATTMs

# A Pair of Complementary ATTMs

- Consider $M_+$ and $M_-$ of the form ATTM[k,(r,0),R] and ATTM[k,(r,1),$R^c$] (where r>1/2)

# A Pair of Complementary ATTMs

- Consider $M_+$ and $M_-$ of the form ATTM[k,(r,0),R] and ATTM[k,(r,1),$R^c$] (where r>1/2)
- We'll call it a pair of complementary (k,r) ATTMs

# A Pair of Complementary ATTMs

- Consider $M_+$ and $M_-$ of the form ATTM[k,(r,0),R] and ATTM[k,(r,1),$R^c$] (where r>1/2)
- We'll call it a pair of complementary (k,r) ATTMs
- For any r>1/2, $\{x|\ M_+(x)=1\}$ and $\{x|\ M_-(x)=1\}$ are disjoint

# A Pair of Complementary ATTMs

- Consider $M_+$ and $M_-$ of the form ATTM[k,(r,0),R] and ATTM[k,(r,1),$R^c$] (where r>1/2)
- We'll call it a pair of complementary (k,r) ATTMs
- For any r>1/2, $\{x|\ M_+(x)=1\}$ and $\{x|\ M_-(x)=1\}$ are disjoint
  - M = ATTM[k,(1-r,1),$R^c$] is the complement of $M_+$: $\{x|\ M_+(x)=0\} = \{x|\ M(x)=1\}$

# A Pair of Complementary ATTMs

- Consider $M_+$ and $M_-$ of the form ATTM[k,(r,0),R] and ATTM[k,(r,1),$R^c$] (where r>1/2)
- We'll call it a pair of complementary (k,r) ATTMs
- For any r>1/2, $\{x|\ M_+(x)=1\}$ and $\{x|\ M_-(x)=1\}$ are disjoint
  - M = ATTM[k,(1-r,1),$R^c$] is the complement of $M_+$: $\{x|\ M_+(x)=0\} = \{x|\ M(x)=1\}$
  - If r > 1-r, $M_-$ stricter than M: $\{x|\ M_-(x)=1\} \subseteq \{x|\ M(x)=1\}$

# A Pair of Complementary ATTMs

# A Pair of Complementary ATTMs

# A Pair of Complementary ATTMs

- L is said to have a pair of complementary ATTMs $(M_+, M_-)$ if

# A Pair of Complementary ATTMs

- L is said to have a pair of complementary ATTMs $(M_+, M_-)$ if
  - $x \in L \Leftrightarrow M_+(x)=1$ and $M_-(x)=0$

# A Pair of Complementary ATTMs

- L is said to have a pair of complementary ATTMs $(M_+, M_-)$ if
  - $x \in L \Leftrightarrow M_+(x)=1$ and $M_-(x)=0$
  - $x \notin L \Leftrightarrow M_+(x)=0$ and $M_-(x)=1$

# A Pair of Complementary ATTMs

- L is said to have a pair of complementary ATTMs $(M_+, M_-)$ if
  - $x \in L \Leftrightarrow M_+(x)=1$ and $M_-(x)=0$
  - $x \notin L \Leftrightarrow M_+(x)=0$ and $M_-(x)=1$
- Exact threshold not critical

# A Pair of Complementary ATTMs

- L is said to have a pair of complementary ATTMs $(M_+, M_-)$ if
  - $x \in L \Leftrightarrow M_+(x)=1$ and $M_-(x)=0$
  - $x \notin L \Leftrightarrow M_+(x)=0$ and $M_-(x)=1$
- Exact threshold not critical
  - Threshold of $(M_+, M_-)$ can be reduced to any $r > 1/2$

# A Pair of Complementary ATTMs

- L is said to have a pair of complementary ATTMs $(M_+, M_-)$ if
  - $x \in L \Leftrightarrow M_+(x)=1$ and $M_-(x)=0$
  - $x \notin L \Leftrightarrow M_+(x)=0$ and $M_-(x)=1$
- Exact threshold not critical
  - Threshold of $(M_+, M_-)$ can be reduced to any $r > 1/2$
    - Reducing threshold enlarges $\{x|\ M_+(x)=1\}$ and $\{x|\ M_-(x)=1\}$

# A Pair of Complementary ATTMs

- L is said to have a pair of complementary ATTMs $(M_+,M_-)$ if
  - $x \in L \Leftrightarrow M_+(x)=1$ and $M_-(x)=0$
  - $x \notin L \Leftrightarrow M_+(x)=0$ and $M_-(x)=1$
- Exact threshold not critical
  - Threshold of $(M_+,M_-)$ can be reduced to any $r > 1/2$
    - Reducing threshold enlarges $\{x|\ M_+(x)=1\}$ and $\{x|\ M_-(x)=1\}$
    - And they stay disjoint

# A Pair of Complementary ATTMs

- L is said to have a pair of complementary ATTMs $(M_+,M_-)$ if
  - $x \in L \Leftrightarrow M_+(x)=1$ and $M_-(x)=0$
  - $x \notin L \Leftrightarrow M_+(x)=0$ and $M_-(x)=1$
- Exact threshold not critical
  - Threshold of $(M_+,M_-)$ can be reduced to any $r > 1/2$
    - Reducing threshold enlarges $\{x|\ M_+(x)=1\}$ and $\{x|\ M_-(x)=1\}$
    - And they stay disjoint
    - So they do not change (as they were already a partitioning)

# A Pair of Complementary ATTMs

- L is said to have a pair of complementary ATTMs $(M_+,M_-)$ if
  - $x \in L \Leftrightarrow M_+(x)=1$ and $M_-(x)=0$
  - $x \notin L \Leftrightarrow M_+(x)=0$ and $M_-(x)=1$
- Exact threshold not critical
  - Threshold of $(M_+,M_-)$ can be reduced to any $r > 1/2$
    - Reducing threshold enlarges $\{x|\ M_+(x)=1\}$ and $\{x|\ M_-(x)=1\}$
    - And they stay disjoint
    - So they do not change (as they were already a partitioning)
  - By parallel repetition, can increase threshold to exponentially close to 1, starting from $1/2 + c$

# AM and ATTM-pairs

# AM and ATTM-pairs

- A language L has an AM[k,r] protocol **iff** L has a pair of complementary (k,r) ATTMs for r>1/2+c

# AM and ATTM-pairs

- A language L has an AM[k,r] protocol **iff** L has a pair of complementary (k,r) ATTMs for r>1/2+c
  - Guarantees on probability of acceptance translated to threshold guarantees, and vice versa

# AM and ATTM-pairs

- A language L has an AM[k,r] protocol **iff** L has a pair of complementary (k,r) ATTMs for r>1/2+c
  - Guarantees on probability of acceptance translated to threshold guarantees, and vice versa
    - AM[k,r] protocol → (k,r′) ATTM pair: natural conversion works if $r > 1-2^{-2k}$ and r′ = 3/4 [Exercise]

# AM and ATTM-pairs

- A language L has an AM[k,r] protocol **iff** L has a pair of complementary (k,r) ATTMs for r>1/2+c
  - Guarantees on probability of acceptance translated to threshold guarantees, and vice versa
    - AM[k,r] protocol → (k,r′) ATTM pair: natural conversion works if $r > 1-2^{-2k}$ and r′ = 3/4 [Exercise]
    - (k,r′) ATTM pair → AM[k,r] protocol: natural conversion works if r′ > 1-1/4k and r = 3/4 [Exercise]

# AM and ATTM-pairs

- A language L has an AM[k,r] protocol **iff** L has a pair of complementary (k,r) ATTMs for r>1/2+c
  - Guarantees on probability of acceptance translated to threshold guarantees, and vice versa
    - AM[k,r] protocol → (k,r′) ATTM pair: natural conversion works if $r > 1-2^{-2k}$ and r′ = 3/4 [Exercise]
    - (k,r′) ATTM pair → AM[k,r] protocol: natural conversion works if r′ > 1-1/4k and r = 3/4 [Exercise]
  - Enough, because we can reduce error (increase thresholds) for both AM protocols and ATTMs

# AM[k] = AM

# AM[k] = AM

- In terms of ATTM-pairs

# AM[k] = AM

- In terms of ATTM-pairs
  - Flipping MA to AM: reduces depth, does not change size, but requires threshold to be reduced from $1 - 1/2^{m+2}$ to $3/4$

# AM[k] = AM

- In terms of ATTM-pairs
  - Flipping MA to AM: reduces depth, does not change size, but requires threshold to be reduced from $1 - 1/2^{m+2}$ to $3/4$
  - Amplifying again: Threshold increased to $1 - 1/2^{m+2}$, but size increased by a polynomial factor

# AM[k] = AM

- In terms of ATTM-pairs
  - Flipping MA to AM: reduces depth, does not change size, but requires threshold to be reduced from $1 - 1/2^{m+2}$ to 3/4
  - Amplifying again: Threshold increased to $1 - 1/2^{m+2}$, but size increased by a polynomial factor
  - Repeat ~k/2 times to reduce to AM[2]

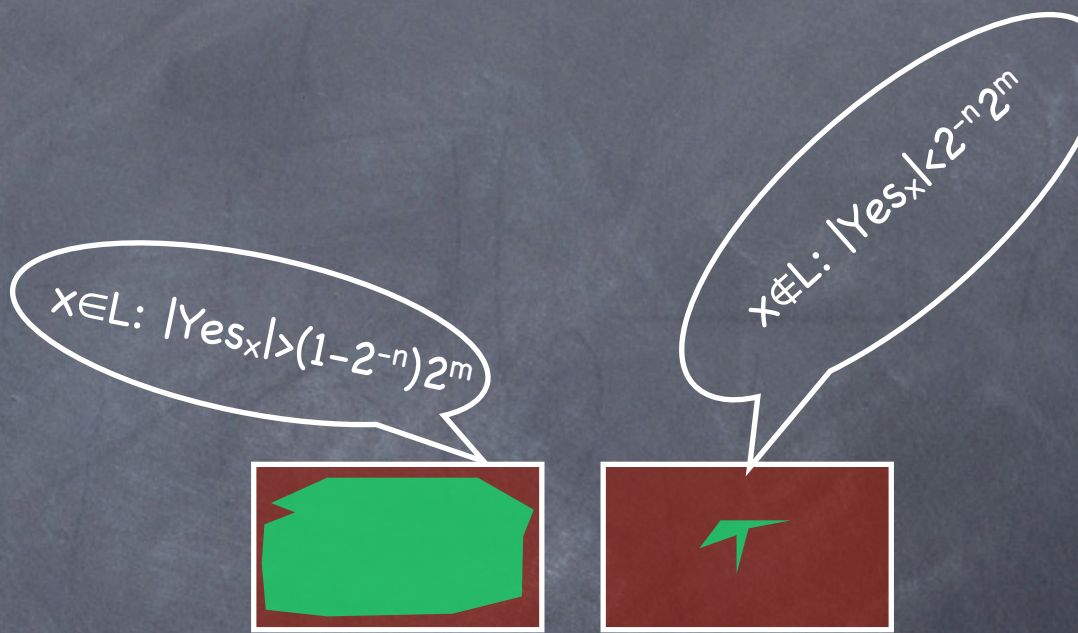
# One-Sided Error

# One-Sided Error

- Recall BPP $\subseteq \Sigma_2^P$

# One-Sided Error

- Recall BPP $\subseteq \Sigma_2^P$

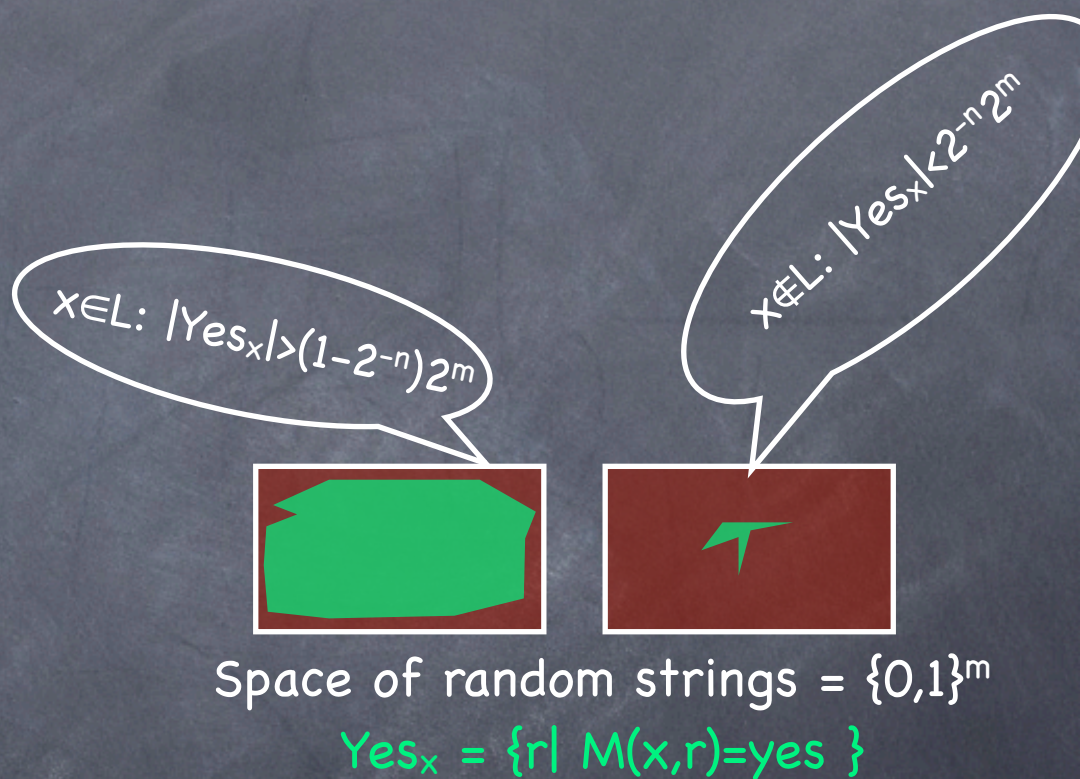



Space of random strings = $\{0,1\}^m$

$Yes_x = \{r |\ M(x,r)=yes\ \}$

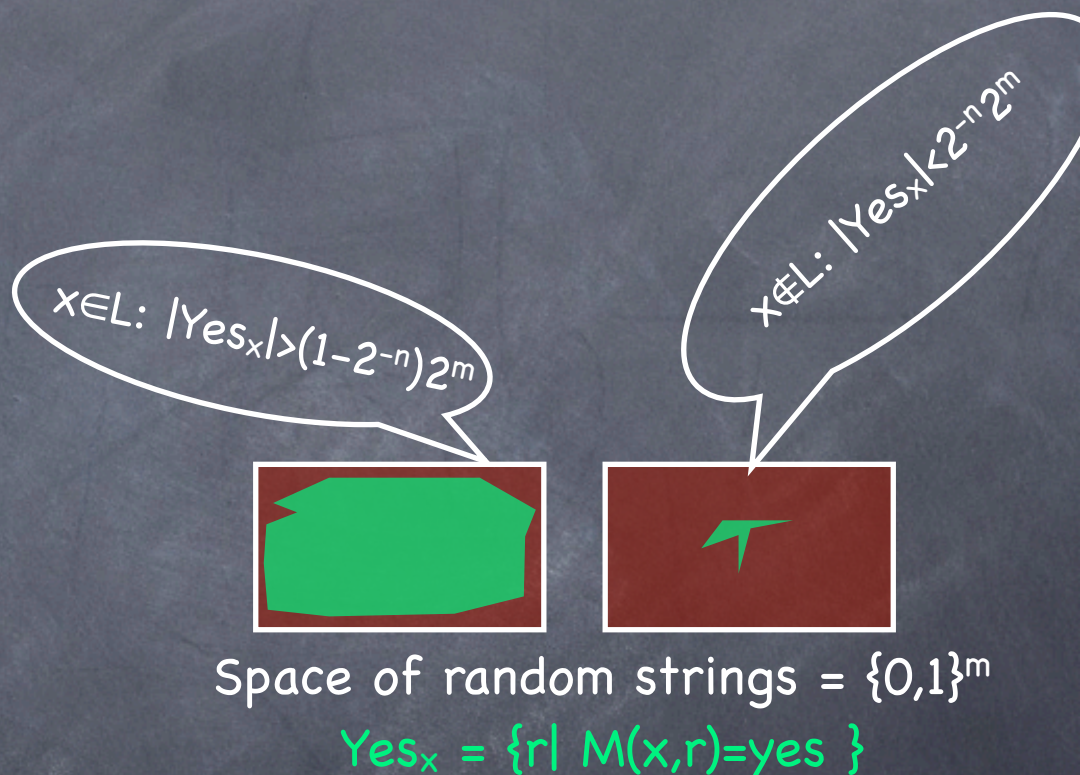
# One-Sided Error

- Recall BPP $\subseteq \Sigma_2^P$
  - Using "shifts" of random tapes



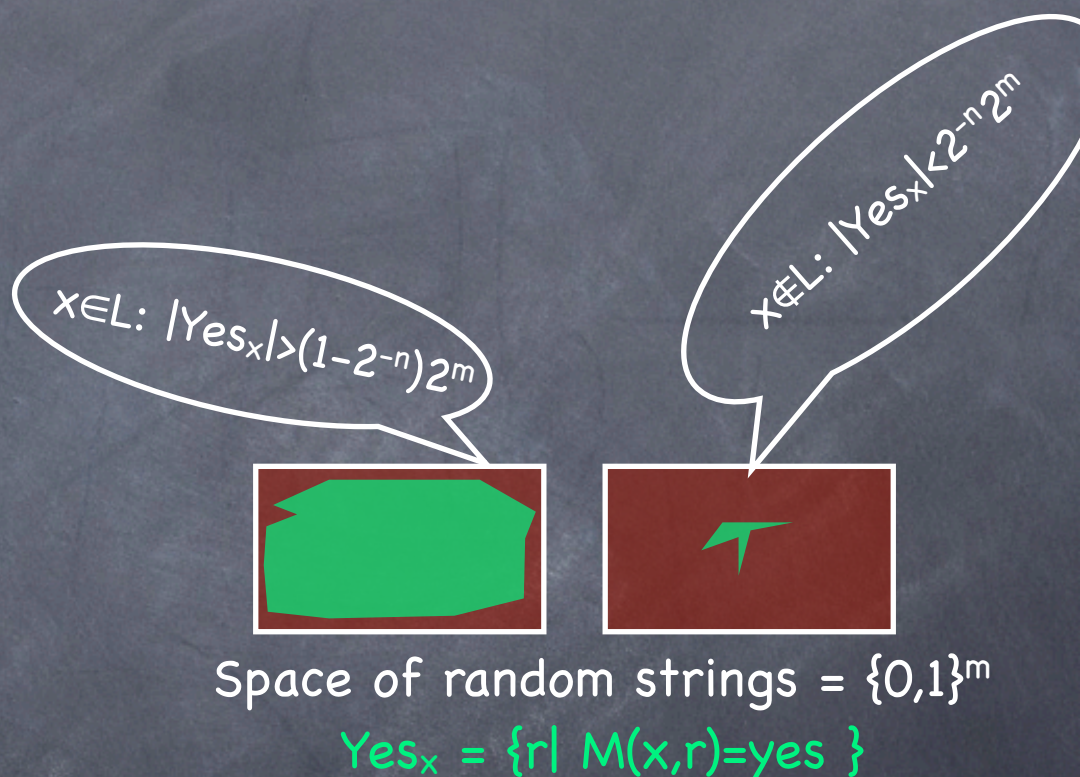
# One-Sided Error

- Recall BPP $\subseteq \Sigma_2^P$
  - Using "shifts" of random tapes
  - $x \in L \Rightarrow \exists P\ P(Yes_x) = \{0,1\}^m$



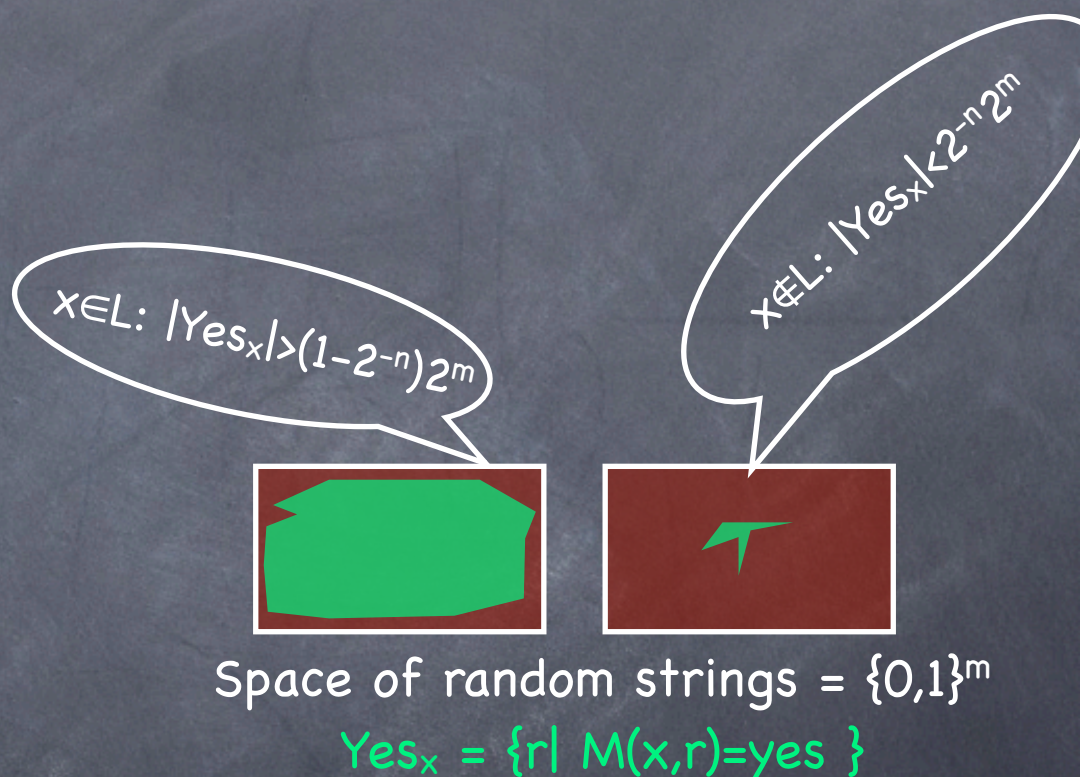
# One-Sided Error

- Recall BPP $\subseteq \Sigma_2^P$
  - Using "shifts" of random tapes
  - $x \in L \Rightarrow \exists P\ P(Yes_x) = \{0,1\}^m$
  - $x \notin L \Rightarrow \forall P\ |P(Yes_x)| < 2^m/4$



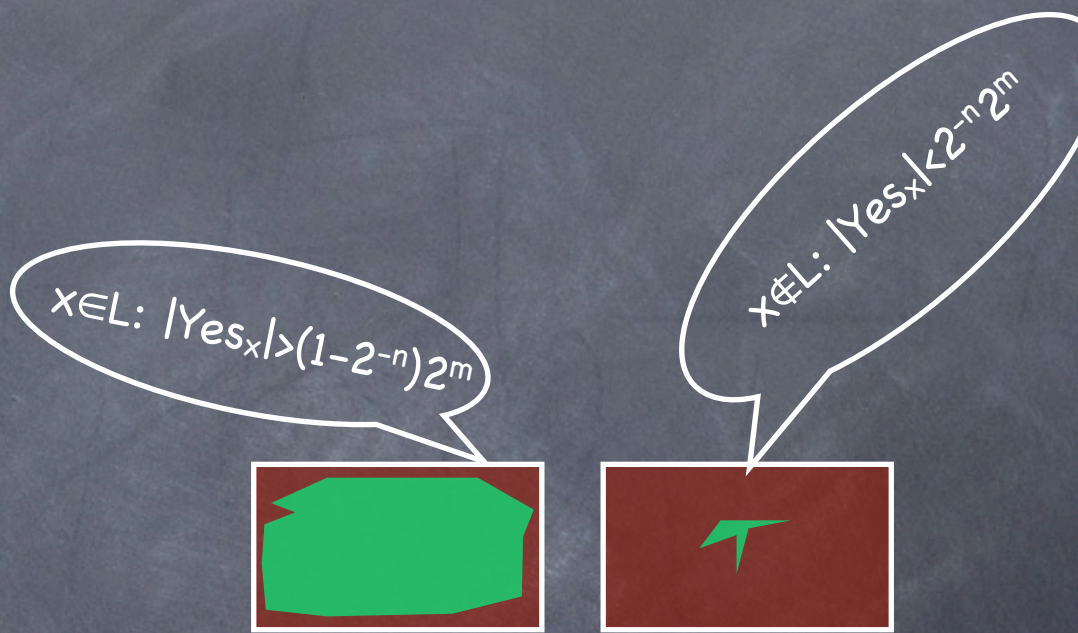
# One-Sided Error

- Recall BPP $\subseteq \Sigma_2^P$
  - Using "shifts" of random tapes
  - $x \in L \Rightarrow \exists P\ P(Yes_x) = \{0,1\}^m$
  - $x \notin L \Rightarrow \forall P\ |P(Yes_x)| < 2^m/4$
- As an MA protocol



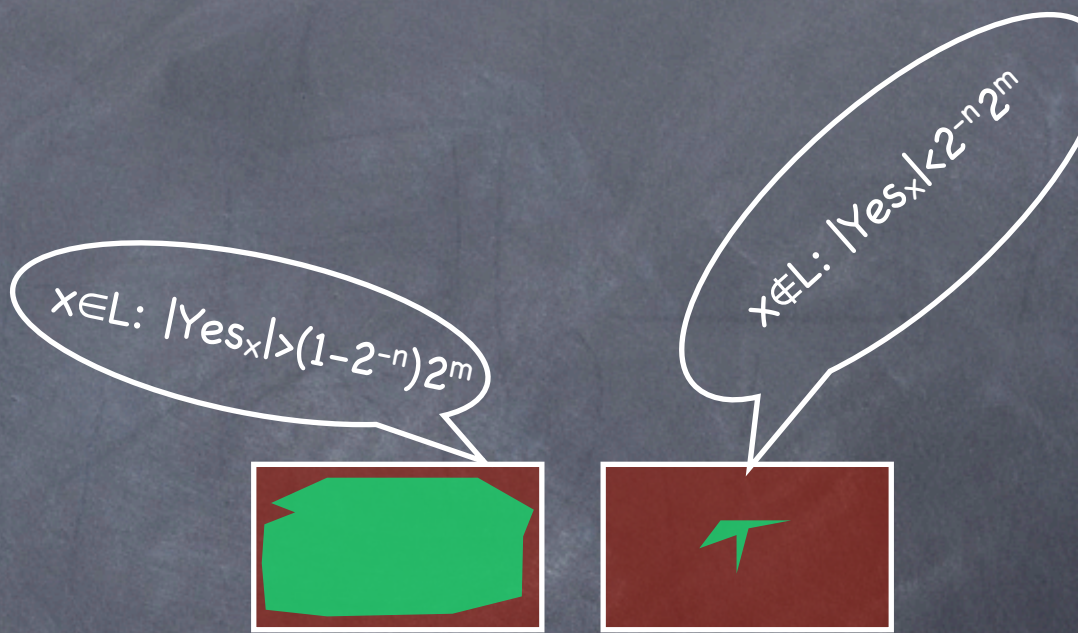
# One-Sided Error

- Recall BPP $\subseteq \Sigma_2^P$
  - Using "shifts" of random tapes
  - $x \in L \Rightarrow \exists P\ P(Yes_x) = \{0,1\}^m$
  - $x \notin L \Rightarrow \forall P\ |P(Yes_x)| < 2^m/4$
- As an MA protocol
  - Merlin sends P



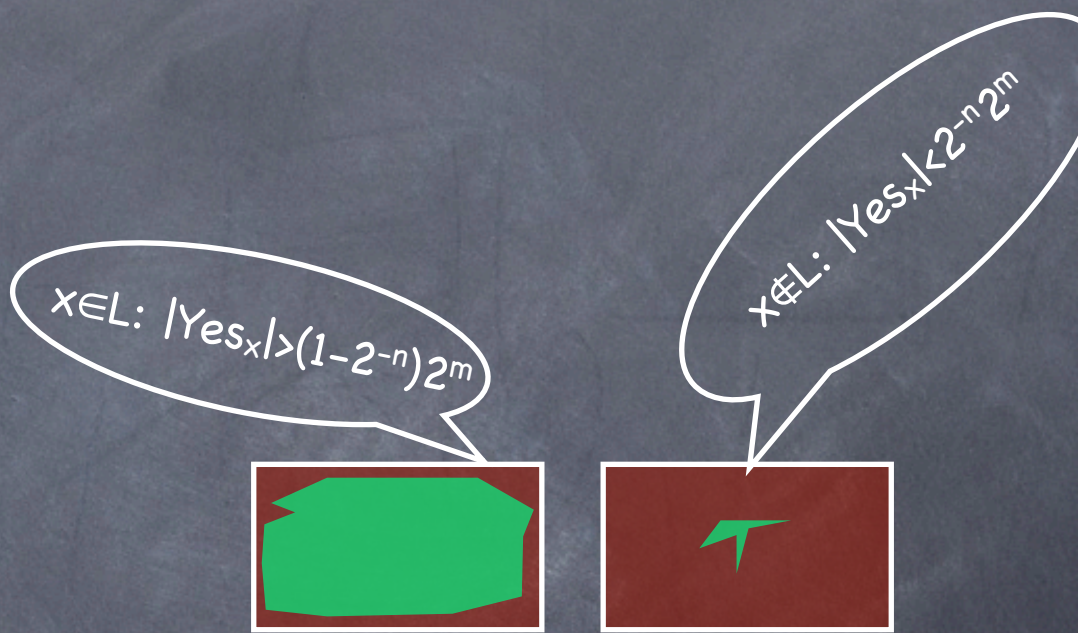
# One-Sided Error

- Recall BPP $\subseteq \Sigma_2^P$
  - Using "shifts" of random tapes
  - $x \in L \Rightarrow \exists P\ P(Yes_x) = \{0,1\}^m$
  - $x \notin L \Rightarrow \forall P\ |P(Yes_x)| < 2^m/4$
- As an MA protocol
  - Merlin sends P
  - Arthur picks $r \leftarrow \{0,1\}^m$



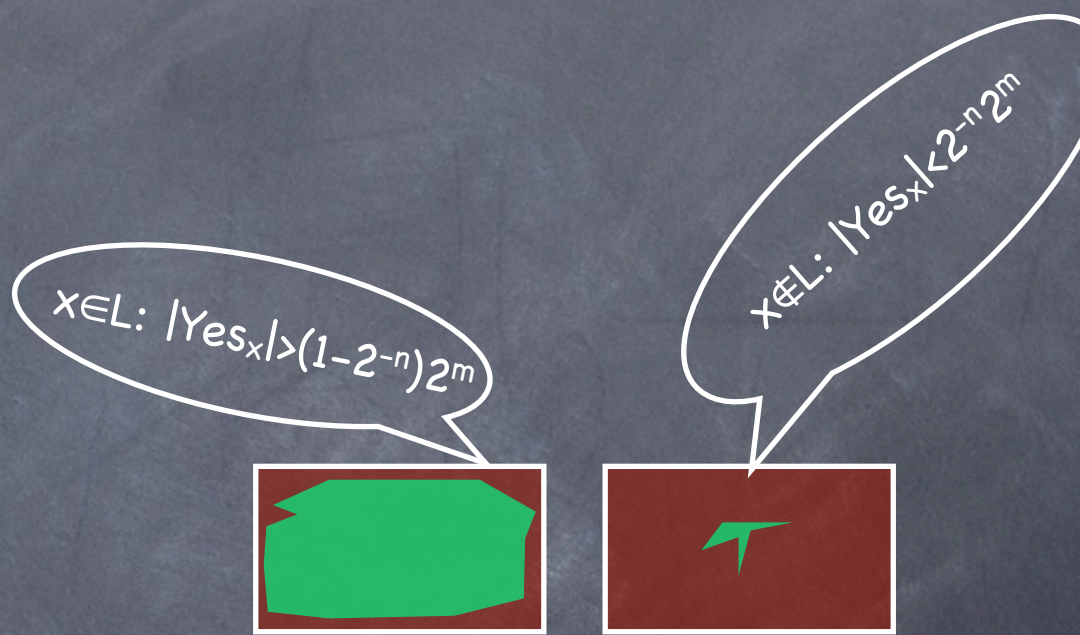
# One-Sided Error

- Recall BPP $\subseteq \Sigma_2^P$
  - Using "shifts" of random tapes
  - $x \in L \Rightarrow \exists P\ P(Yes_x) = \{0,1\}^m$
  - $x \notin L \Rightarrow \forall P\ |P(Yes_x)| < 2^m/4$
- As an MA protocol
  - Merlin sends P
  - Arthur picks $r \leftarrow \{0,1\}^m$
  - Merlin sends $s \in Yes_x$ s.t. $r \in P(s)$



# One-Sided Error

- Recall BPP $\subseteq \Sigma_2^P$
  - Using "shifts" of random tapes
  - $x \in L \Rightarrow \exists P\ P(Yes_x) = \{0,1\}^m$
  - $x \notin L \Rightarrow \forall P\ |P(Yes_x)| < 2^m/4$
- As an MA protocol
  - Merlin sends P
  - Arthur picks $r \leftarrow \{0,1\}^m$
  - Merlin sends $s \in Yes_x$ s.t. $r \in P(s)$
- One-sided error




Space of random strings = $\{0,1\}^m$

$Yes_x = \{r |\ M(x,r)=yes\ \}$

# Perfect Completeness

# Perfect Completeness

- Converting MA protocol to perfectly complete MA

# Perfect Completeness

- Converting MA protocol to perfectly complete MA
  - Consider $Yes_{x,a}$ where a is the message from Merlin

# Perfect Completeness

- Converting MA protocol to perfectly complete MA
  - Consider $Yes_{x,a}$ where a is the message from Merlin
  - $x \in L \Rightarrow \exists a, P \;\; P(Yes_{x,a}) = \{0,1\}^m$

# Perfect Completeness

- Converting MA protocol to perfectly complete MA
  - Consider $Yes_{x,a}$ where a is the message from Merlin
  - $x \in L \Rightarrow \exists a, P \quad P(Yes_{x,a}) = \{0,1\}^m$
  - $x \notin L \Rightarrow \forall a, P \quad |P(Yes_{x,a})| < 2^m/4$

# Perfect Completeness

- Converting MA protocol to perfectly complete MA
  - Consider $Yes_{x,a}$ where $a$ is the message from Merlin
  - $x \in L \Rightarrow \exists a, P \;\; P(Yes_{x,a}) = \{0,1\}^m$
  - $x \notin L \Rightarrow \forall a, P \;\; |P(Yes_{x,a})| < 2^m/4$
- Perfectly complete MA protocol

# Perfect Completeness

- Converting MA protocol to perfectly complete MA
  - Consider $Yes_{x,a}$ where $a$ is the message from Merlin
  - $x \in L \Rightarrow \exists a,P \;\; P(Yes_{x,a}) = \{0,1\}^m$
  - $x \notin L \Rightarrow \forall a,P \;\; |P(Yes_{x,a})| < 2^m/4$
- Perfectly complete MA protocol
  - Merlin sends $a$, $P$

# Perfect Completeness

- Converting MA protocol to perfectly complete MA
  - Consider $Yes_{x,a}$ where $a$ is the message from Merlin
  - $x \in L \Rightarrow \exists a,P \;\; P(Yes_{x,a}) = \{0,1\}^m$
  - $x \notin L \Rightarrow \forall a,P \;\; |P(Yes_{x,a})| < 2^m/4$
- Perfectly complete MA protocol
  - Merlin sends $a$, $P$
  - Arthur picks $r \leftarrow \{0,1\}^m$

# Perfect Completeness

- Converting MA protocol to perfectly complete MA
  - Consider $Yes_{x,a}$ where $a$ is the message from Merlin
  - $x \in L \Rightarrow \exists a,P \;\; P(Yes_{x,a}) = \{0,1\}^m$
  - $x \notin L \Rightarrow \forall a,P \;\; |P(Yes_{x,a})| < 2^m/4$
- Perfectly complete MA protocol
  - Merlin sends $a$, $P$
  - Arthur picks $r \leftarrow \{0,1\}^m$
    - Checks if there exists $s \in P^{-1}(r)$ s.t. $s \in Yes_{x,a}$

# Perfect Completeness

- Converting MA protocol to perfectly complete MA
  - Consider $Yes_{x,a}$ where $a$ is the message from Merlin
  - $x \in L \Rightarrow \exists a,P \;\; P(Yes_{x,a}) = \{0,1\}^m$
  - $x \notin L \Rightarrow \forall a,P \;\; |P(Yes_{x,a})| < 2^m/4$
- Perfectly complete MA protocol
  - Merlin sends $a$, $P$
  - Arthur picks $r \leftarrow \{0,1\}^m$
    - Checks if there exists $s \in P^{-1}(r)$ s.t. $s \in Yes_{x,a}$
- Converting AM protocols

# Perfect Completeness

- Converting MA protocol to perfectly complete MA
  - Consider $Yes_{x,a}$ where $a$ is the message from Merlin
  - $x \in L \Rightarrow \exists a,P \;\; P(Yes_{x,a}) = \{0,1\}^m$
  - $x \notin L \Rightarrow \forall a,P \;\; |P(Yes_{x,a})| < 2^m/4$
- Perfectly complete MA protocol
  - Merlin sends $a$, $P$
  - Arthur picks $r \leftarrow \{0,1\}^m$
    - Checks if there exists $s \in P^{-1}(r)$ s.t. $s \in Yes_{x,a}$
- Converting AM protocols
  - $Yes_x = \{r |$ $\exists a$ s.t. Arthur accepts $x$ on transcript $(r,a)\}$

# Perfect Completeness

- Converting MA protocol to perfectly complete MA
  - Consider $Yes_{x,a}$ where a is the message from Merlin
  - $x \in L \Rightarrow \exists a,P \;\; P(Yes_{x,a}) = \{0,1\}^m$
  - $x \notin L \Rightarrow \forall a,P \;\; |P(Yes_{x,a})| < 2^m/4$
- Perfectly complete MA protocol
  - Merlin sends a, P
  - Arthur picks $r \leftarrow \{0,1\}^m$
    - Checks if there exists $s \in P^{-1}(r)$ s.t. $s \in Yes_{x,a}$
- Converting AM protocols
  - $Yes_x = \{r |$ $\exists a$ s.t. Arthur accepts x on transcript (r,a) $\}$
  - A one-sided error MAM protocol: (P, r, a)

# Perfect Completeness

- Converting MA protocol to perfectly complete MA
  - Consider $Yes_{x,a}$ where a is the message from Merlin
  - $x \in L \Rightarrow \exists a,P \;\; P(Yes_{x,a}) = \{0,1\}^m$
  - $x \notin L \Rightarrow \forall a,P \;\; |P(Yes_{x,a})| < 2^m/4$
- Perfectly complete MA protocol
  - Merlin sends a, P
  - Arthur picks $r \leftarrow \{0,1\}^m$
    - Checks if there exists $s \in P^{-1}(r)$ s.t. $s \in Yes_{x,a}$
- Converting AM protocols
  - $Yes_x = \{r |$ $\exists a$ s.t. Arthur accepts x on transcript (r,a) $\}$
  - A one-sided error MAM protocol: (P, r, a)
  - But MAM = AM (and preserves completeness)

# Perfect Completeness

# Perfect Completeness

- Therefore requiring perfect completeness does not change the classes MA or AM

# Perfect Completeness

- Therefore requiring perfect completeness does not change the classes MA or AM
  - Contrast with RP vs. BPP

# Today

# Today

- MA ⊆ AM. MAM = AM.

# Today

- MA ⊆ AM. MAM = AM.
- AM[k] = AM for k ≥ 2

# Today

- MA ⊆ AM. MAM = AM.
- AM[k] = AM for k ≥ 2
  - Using alternate characterization in terms of pairs of complementary ATTMs

# Today

- MA ⊆ AM. MAM = AM.
- AM[k] = AM for k ≥ 2
  - Using alternate characterization in terms of pairs of complementary ATTMs
- one-sided-error-AM = AM

# Today

- MA ⊆ AM. MAM = AM.
- AM[k] = AM for k ≥ 2
  - Using alternate characterization in terms of pairs of complementary ATTMs
- one-sided-error-AM = AM
- Coming up:

# Today

- MA ⊆ AM. MAM = AM.
- AM[k] = AM for $k \geq 2$
  - Using alternate characterization in terms of pairs of complementary ATTMs
- one-sided-error-AM = AM
- Coming up:
  - A little more of AM (and where it fits into the zoo)

# Today

- MA ⊆ AM. MAM = AM.
- AM[k] = AM for k ≥ 2
  - Using alternate characterization in terms of pairs of complementary ATTMs
- one-sided-error-AM = AM
- Coming up:
  - A little more of AM (and where it fits into the zoo)
  - Some other concepts in interactive proofs