Experts/Zero-Sum Games Equilibrium.
Today

Experts/Zero-Sum Games Equilibrium.
Boosting and Experts.
Today

Experts/Zero-Sum Games Equilibrium.
Boosting and Experts.
Routing and Experts.
Experts/Zero-Sum Games Equilibrium.
Boosting and Experts.
Routing and Experts.
Linear Programming Introduction (Gentle)
Games and experts

Again: find \((x^*, y^*)\), such that
Games and experts

Again: find \((x^*, y^*)\), such that

\[
\left( \max_y x^* A y \right) - \left( \min_x x^* A y^* \right) \leq \varepsilon
\]
Games and experts

Again: find \((x^*, y^*)\), such that

\[
(\max_y x^* Ay) - (\min_x x^* Ay^*) \leq \varepsilon
\]

\[
C(x^*) - R(y^*) \leq \varepsilon
\]
Games and experts

Again: find \((x^*, y^*)\), such that

\[
(max_y x^* A y) - (min_x x^* A y^*) \leq \varepsilon
\]

\[
C(x^*) - R(y^*) \leq \varepsilon
\]
Games and experts

Again: find \((x^*, y^*)\), such that

\[
\max_y x^* Ay - \min_x x^* Ay^* \leq \varepsilon
\]

\[
C(x^*) - R(y^*) \leq \varepsilon
\]

Experts Framework:

\(n\) Experts,
Games and experts

Again: find \((x^*, y^*)\), such that

\[
(\max_y x^* Ay) - (\min_x x^* Ay^*) \leq \varepsilon
\]

\[
C(x^*) - R(y^*) \leq \varepsilon
\]

Experts Framework:

\(n\) Experts, \(T\) days,
Games and experts

Again: find \((x^*, y^*)\), such that
\[
(\max_y x^* Ay) - (\min_x x^* Ay^*) \leq \varepsilon
\]
\[
C(x^*) - R(y^*) \leq \varepsilon
\]

Experts Framework:
\(n\) Experts, \(T\) days, \(L^*\) -total loss of best expert.
Games and experts

Again: find \((x^*, y^*)\), such that
\[
\begin{align*}
(x^* A y) - (x^* A y^*) & \leq \varepsilon \\
C(x^*) - R(y^*) & \leq \varepsilon
\end{align*}
\]

Experts Framework:
\(n\) Experts, \(T\) days, \(L^*\) -total loss of best expert.

Multiplicative Weights Method yields loss \(L\) where
Games and experts

Again: find \((x^*, y^*)\), such that

\[
\left( \max_y x^* Ay \right) - \left( \min_x x^* Ay^* \right) \leq \varepsilon
\]

\[
C(x^*) - R(y^*) \leq \varepsilon
\]

Experts Framework:

\(n\) Experts, \(T\) days, \(L^*\) - total loss of best expert.

Multiplicative Weights Method yields loss \(L\) where

\[
L \leq (1 + \varepsilon)L^* + \frac{\log n}{\varepsilon}
\]
Games and Experts.

Assume: $A$ has payoffs in $[0, 1]$. 
Assume: $A$ has payoffs in $[0, 1]$.

For $T = \frac{\log n}{\varepsilon^2}$ days:
Games and Experts.

Assume: $A$ has payoffs in $[0, 1]$.

For $T = \frac{\log n}{\epsilon^2}$ days:

1) $m$ pure row strategies are experts.
Assume: $A$ has payoffs in $[0, 1]$.

For $T = \frac{\log n}{\varepsilon^2}$ days:

1) $m$ pure row strategies are experts.
Use multiplicative weights, produce row distribution.
Assume: $A$ has payoffs in $[0, 1]$.

For $T = \frac{\log n}{\varepsilon^2}$ days:

1) $m$ pure row strategies are experts.
Use multiplicative weights, produce row distribution.
Let $x_t$ be distribution (row strategy) $x_t$ on day $t$. 
Games and Experts.

Assume: $A$ has payoffs in $[0, 1]$.

For $T = \frac{\log n}{\varepsilon^2}$ days:

1) $m$ pure row strategies are experts.
Use multiplicative weights, produce row distribution.
Let $x_t$ be distribution (row strategy) $x_t$ on day $t$.

2) Each day, adversary plays best column response to $x_t$.
Let $y_t$ be indicator vector for this column.
Let $y^* = \frac{1}{T} \sum t y_t$ and $x^* = \arg\min_{x_t} x_t A y_t$.

Assume: $A$ has payoffs in $[0, 1]$.

For $T = \frac{\log n}{\varepsilon^2}$ days:

1) $m$ pure row strategies are experts.
   Use multiplicative weights, produce row distribution.
   Let $x_t$ be distribution (row strategy) $x_t$ on day $t$.

2) Each day, adversary plays best column response to $x_t$.
   Choose column of $A$ that maximizes row’s expected loss.
Assume: $A$ has payoffs in $[0, 1]$.

For $T = \frac{\log n}{\varepsilon^2}$ days:

1) $m$ pure row strategies are experts.
Use multiplicative weights, produce row distribution.
Let $x_t$ be distribution (row strategy) $x_t$ on day $t$.

2) Each day, adversary plays best column response to $x_t$.
Choose column of $A$ that maximizes row’s expected loss.
Let $y_t$ be indicator vector for this column.
Assume: $A$ has payoffs in $[0, 1]$.

For $T = \frac{\log n}{\epsilon^2}$ days:

1) $m$ pure row strategies are experts.
Use multiplicative weights, produce row distribution.
Let $x_t$ be distribution (row strategy) $x_t$ on day $t$.

2) Each day, adversary plays best column response to $x_t$.
Choose column of $A$ that maximizes row’s expected loss.
Let $y_t$ be indicator vector for this column.

Let $y^* = \frac{1}{T} \sum_t y_t$
Assume: $A$ has payoffs in $[0, 1]$.

For $T = \frac{\log n}{\varepsilon^2}$ days:

1) $m$ pure row strategies are experts.
   
   Use multiplicative weights, produce row distribution.
   
   Let $x_t$ be distribution (row strategy) $x_t$ on day $t$.

2) Each day, adversary plays best column response to $x_t$.
   
   Choose column of $A$ that maximizes row’s expected loss.
   
   Let $y_t$ be indicator vector for this column.

   Let $y^* = \frac{1}{T} \sum_t y_t$ and $x^* = \arg\min_{x_t} x_t A y_t$. 
Assume: $A$ has payoffs in $[0, 1]$.

For $T = \frac{\log n}{\epsilon^2}$ days:

1) $m$ pure row strategies are experts. 
Use multiplicative weights, produce row distribution. 
Let $x_t$ be distribution (row strategy) $x_t$ on day $t$.

2) Each day, adversary plays best column response to $x_t$. 
Choose column of $A$ that maximizes row’s expected loss. 
Let $y_t$ be indicator vector for this column. 
Let $y^* = \frac{1}{T} \sum t y_t$ and $x^* = \arg\min_{x_t} x_t A y_t$. 
Let $y^* = \frac{1}{T} \sum t y_t$
Assume: $A$ has payoffs in $[0, 1]$.

For $T = \frac{\log n}{\varepsilon^2}$ days:

1) $m$ pure row strategies are experts.

Use multiplicative weights, produce row distribution. Let $x_t$ be distribution (row strategy) $x_t$ on day $t$.

2) Each day, adversary plays best column response to $x_t$. Choose column of $A$ that maximizes row’s expected loss. Let $y_t$ be indicator vector for this column.

Let $y^* = \frac{1}{T} \sum_t y_t$ and $x^* = \arg\min_{x_t} x_t A y_t$.

Let $y^* = \frac{1}{T} \sum_t y_t$ and $x^* = \frac{1}{T} \sum_t x_t$. 
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$. 
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$. 
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.
Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$.

Claim: $(x^*, y^*)$ are $2\varepsilon$-optimal for matrix $A$. 
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$.

Claim: $(x^*, y^*)$ are $2\varepsilon$-optimal for matrix $A$.

Column payoff: $C(x^*) = \max_y x^* A y$. 
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$.

Claim: $(x^*, y^*)$ are $2\varepsilon$-optimal for matrix $A$.

Column payoff: $C(x^*) = \max_y x^* A y$.

Let $y_r$ be best response to $C(x^*)$. 
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$.

**Claim:** $(x^*, y^*)$ are $2\varepsilon$-optimal for matrix $A$.

Column payoff: $C(x^*) = \max_y x^* Ay$.
   - Let $y_r$ be best response to $C(x^*)$.
   - Day $t$, $x_t Ay_t \geq x_t Ay_r$. Since $y_t$ is best response to $x_t$. 
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$.

**Claim:** $(x^*, y^*)$ are $2\varepsilon$-optimal for matrix $A$.

Column payoff: $C(x^*) = \max_y x^* A y$.

Let $y_r$ be best response to $C(x^*)$.

Day $t$, $x_t A y_t \geq x_t A y_r$. Since $y_t$ is best response to $x_t$.

Algorithm loss: $\sum_t x_t A y_t \geq \sum_t x_t A y_r$.
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$.

Claim: $(x^*, y^*)$ are $2\varepsilon$-optimal for matrix $A$.

Column payoff: $C(x^*) = \max_y x^* A y$.

Let $y_r$ be best response to $C(x^*)$.

Day $t$, $x_t A y_t \geq x_t A y_r$. Since $y_t$ is best response to $x_t$.

Algorithm loss: $\sum_t x_t A y_t \geq \sum_t x_t A y_r$

$L \geq T \times C(x^*)$. 

Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$.

Claim: $(x^*, y^*)$ are $2\varepsilon$-optimal for matrix $A$.

Column payoff: $C(x^*) = \max_y x^* Ay$.

Let $y_r$ be best response to $C(x^*)$.

Day $t$, $x_tA_y \geq x_tA_{y_r}$. Since $y_t$ is best response to $x_t$.

Algorithm loss: $\sum_t x_tA_y \geq \sum_t x_tA_{y_r}$

$L \geq T \times C(x^*)$.

Best expert: $L^*$- best row against all the columns played.
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum x_t$ and $y^* = \frac{1}{T} \sum y_t$.

Claim: $(x^*, y^*)$ are $2\epsilon$-optimal for matrix $A$.

Column payoff: $C(x^*) = \max_y x^* Ay$.

Let $y_r$ be best response to $C(x^*)$.

Day $t$, $x_t Ay_t \geq x_t Ay_r$. Since $y_t$ is best response to $x_t$.

Algorithm loss: $\sum_t x_t Ay_t \geq \sum_t x_t Ay_r$.

$L \geq T \times C(x^*)$.

Best expert: $L^*$- best row against all the columns played.

best row against $\sum_t Ay_t$
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$.

Claim: $(x^*, y^*)$ are $2\varepsilon$-optimal for matrix $A$.

Column payoff: $C(x^*) = \max_y x^* A y$.

Let $y_r$ be best response to $C(x^*)$.

Day $t$, $x_t Ay_t \geq x_t Ay_r$. Since $y_t$ is best response to $x_t$.

Algorithm loss: $\sum_t x_t Ay_t \geq \sum_t x_t Ay_r$

$L \geq T \times C(x^*)$.

Best expert: $L^*$- best row against all the columns played.

best row against $\sum_t Ay_t$ and $Ty^* = \sum_t y_t$
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$.

Claim: $(x^*, y^*)$ are $2\varepsilon$-optimal for matrix $A$.

Column payoff: $C(x^*) = \max_y x^* A y$.

Let $y_r$ be best response to $C(x^*)$.

Day $t$, $x_t A y_t \geq x_t A y_r$. Since $y_t$ is best response to $x_t$.

Algorithm loss: $\sum_t x_t A y_t \geq \sum_t x_t A y_r$

$L \geq T \times C(x^*)$.

Best expert: $L^*$- best row against all the columns played.

best row against $\sum_t A y_t$ and $T y^* = \sum_t y_t$

$\rightarrow$ best row against $T A y^*$.
Approximate Equilibrium: slightly different!

Experts: \( x_t \) is strategy on day \( t \), \( y_t \) is best column against \( x_t \).

Let \( x^* = \frac{1}{T} \sum_t x_t \) and \( y^* = \frac{1}{T} \sum_t y_t \).

**Claim:** \((x^*, y^*)\) are \(2\varepsilon\)-optimal for matrix \( A \).

**Column payoff:** \( C(x^*) = \max_y x^* A y \).

Let \( y_r \) be best response to \( C(x^*) \).

Day \( t \), \( x_t A y_t \geq x_t A y_r \). Since \( y_t \) is best response to \( x_t \).

Algorithm loss: \( \sum_t x_t A y_t \geq \sum_t x_t A y_r \)

\[ L \geq T \times C(x^*). \]

Best expert: \( L^*\)- best row against all the columns played.

best row against \( \sum_t A y_t \) and \( T y^* = \sum_t y_t \)

\( \rightarrow \) best row against \( TA y^* \).

\( \rightarrow L^* \leq T \times R(y^*). \)
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$.

**Claim:** $(x^*, y^*)$ are $2\varepsilon$-optimal for matrix $A$.

Column payoff: $C(x^*) = \max_y x^* Ay$.

Let $y_r$ be best response to $C(x^*)$.

Day $t$, $x_t Ay_t \geq x_t Ay_r$. Since $y_t$ is best response to $x_t$.

Algorithm loss: $\sum_t x_t Ay_t \geq \sum_t x_t Ay_r$

$L \geq T \times C(x^*)$.

Best expert: $L^*$- best row against all the columns played.

best row against $\sum_t Ay_t$ and $Ty^* = \sum_t y_t$

$\rightarrow$ best row against $TAy^*$.

$\rightarrow L^* \leq T \times R(y^*)$. 
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$.

Claim: $(x^*, y^*)$ are $2\varepsilon$-optimal for matrix $A$.

Column payoff: $C(x^*) = \max_y x^* A y$.
Let $y_r$ be best response to $C(x^*)$.
Day $t$, $x_t A y_t \geq x_t A y_r$. Since $y_t$ is best response to $x_t$.
Algorithm loss: $\sum_t x_t A y_t \geq \sum_t x_t A y_r$
$L \geq T \times C(x^*)$.

Best expert: $L^*$- best row against all the columns played.

best row against $\sum_t A y_t$ and $T y^* = \sum_t y_t$
$\rightarrow$ best row against $T A y^*$.
$\rightarrow L^* \leq T \times R(y^*)$.

Multiplicative Weights:
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$.

Claim: $(x^*, y^*)$ are $2\varepsilon$-optimal for matrix $A$.

Column payoff: $C(x^*) = \max_y x^* Ay$.

Let $y_r$ be best response to $C(x^*)$.

Day $t$, $x_t A y_t \geq x_t A y_r$. Since $y_t$ is best response to $x_t$.

Algorithm loss: $\sum_t x_t A y_t \geq \sum_t x_t A y_r$

$L \geq T \times C(x^*)$.

Best expert: $L^*$- best row against all the columns played.

best row against $\sum_t A y_t$ and $T y^* = \sum_t y_t$
$
\rightarrow$ best row against $T A y^*$.

$\rightarrow L^* \leq T \times R(y^*)$.

Multiplicative Weights: $L \leq (1 + \varepsilon)L^* + \frac{\ln n}{\varepsilon}$
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$.

Claim: $(x^*, y^*)$ are $2\varepsilon$-optimal for matrix $A$.

Column payoff: $C(x^*) = \max_y x^* Ay$.

Let $y_r$ be best response to $C(x^*)$.

Day $t$, $x_t Ay_t \geq x_t Ay_r$. Since $y_t$ is best response to $x_t$.

Algorithm loss: $\sum_t x_t Ay_t \geq \sum_t x_t Ay_r$

$L \geq T \times C(x^*)$.

Best expert: $L^*$- best row against all the columns played.

best row against $\sum_t Ay_t$ and $Ty^* = \sum_t y_t$

$\rightarrow$ best row against $TAy^*$.

$\rightarrow L^* \leq T \times R(y^*)$.

Multiplicative Weights: $L \leq (1 + \varepsilon)L^* + \frac{\ln n}{\varepsilon}$

$TC(x^*) \leq (1 + \varepsilon)TR(y^*) + \frac{\ln n}{\varepsilon}$
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$.

Claim: $(x^*, y^*)$ are $2\varepsilon$-optimal for matrix $A$.

Column payoff: $C(x^*) = \max_y x^* A y$.

Let $y_r$ be best response to $C(x^*)$.

Day $t$, $x_t A y_t \geq x_t A y_r$. Since $y_t$ is best response to $x_t$.

Algorithm loss: $\sum_t x_t A y_t \geq \sum_t x_t A y_r$

$L \geq T \times C(x^*)$.

Best expert: $L^*$- best row against all the columns played.

best row against $\sum_t A y_t$ and $Ty^* = \sum_t y_t$

$\rightarrow$ best row against $TAy^*$.

$\rightarrow L^* \leq T \times R(y^*)$.

Multiplicative Weights: $L \leq (1 + \varepsilon)L^* + \frac{\ln n}{\varepsilon}$

$TC(x^*) \leq (1 + \varepsilon) TR(y^*) + \frac{\ln n}{\varepsilon} \rightarrow C(x^*) \leq (1 + \varepsilon) R(y^*) + \frac{\ln n}{\varepsilon T}$
Approximate Equilibrium: slightly different!

Experts: \( x_t \) is strategy on day \( t \), \( y_t \) is best column against \( x_t \).

Let \( x^* = \frac{1}{T} \sum_t x_t \) and \( y^* = \frac{1}{T} \sum_t y_t \).

Claim: \((x^*, y^*)\) are \(2\varepsilon\)-optimal for matrix \( A\).

Column payoff: \( C(x^*) = \max_y x^* Ay \).

Let \( y_r \) be best response to \( C(x^*) \).

Day \( t \), \( x_t Ay_t \geq x_t Ay_r \). Since \( y_t \) is best response to \( x_t \).

Algorithm loss: \( \sum_t x_t Ay_t \geq \sum_t x_t Ay_r \)

\[ L \geq T \times C(x^*). \]

Best expert: \( L^* \)- best row against all the columns played.

\[ \text{best row against } \sum_t Ay_t \text{ and } Ty^* = \sum_t y_t \]

\[ \rightarrow \text{best row against } TAy^*. \]

\[ \rightarrow L^* \leq T \times R(y^*). \]

Multiplicative Weights: \( L \leq (1 + \varepsilon)L^* + \frac{\ln n}{\varepsilon} \)

\[ TC(x^*) \leq (1 + \varepsilon)TR(y^*) + \frac{\ln n}{\varepsilon} \rightarrow C(x^*) \leq (1 + \varepsilon)R(y^*) + \frac{\ln n}{\varepsilon T} \]

\[ \rightarrow C(x^*) - R(y^*) \leq \varepsilon R(y^*) + \frac{\ln n}{\varepsilon T}. \]
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$.

Claim: $(x^*, y^*)$ are $2\varepsilon$-optimal for matrix $A$.

Column payoff: $C(x^*) = \max_y x^* Ay$.

Let $y_r$ be best response to $C(x^*)$.

Day $t$, $x_t Ay_t \geq x_t Ay_r$. Since $y_t$ is best response to $x_t$.

Algorithm loss: $\sum_t x_t Ay_t \geq \sum_t x_t Ay_r$

$L \geq T \times C(x^*)$.

Best expert: $L^*$- best row against all the columns played.

- best row against $\sum_t Ay_t$ and $Ty^* = \sum_t y_t$
- $\rightarrow$ best row against $TAy^*$.
- $\rightarrow L^* \leq T \times R(y^*)$.

Multiplicative Weights: $L \leq (1 + \varepsilon)L^* + \frac{\ln n}{\varepsilon}$

$TC(x^*) \leq (1 + \varepsilon)TR(y^*) + \frac{\ln n}{\varepsilon} \rightarrow C(x^*) \leq (1 + \varepsilon)R(y^*) + \frac{\ln n}{\varepsilon T}$

$\rightarrow C(x^*) - R(y^*) \leq \varepsilon R(y^*) + \frac{\ln n}{\varepsilon T}$.

$T = \frac{\ln n}{\varepsilon^2}, R(y^*) \leq 1$
Approximate Equilibrium: slightly different!

Experts: $x_t$ is strategy on day $t$, $y_t$ is best column against $x_t$.

Let $x^* = \frac{1}{T} \sum_t x_t$ and $y^* = \frac{1}{T} \sum_t y_t$.

Claim: $(x^*, y^*)$ are $2\varepsilon$-optimal for matrix $A$.

Column payoff: $C(x^*) = \max_y x^* Ay$.

Let $y_r$ be best response to $C(x^*)$.

Day $t$, $x_t Ay_t \geq x_t Ay_r$. Since $y_t$ is best response to $x_t$.

Algorithm loss: $\sum_t x_t Ay_t \geq \sum_t x_t Ay_r$

$L \geq T \times C(x^*)$.

Best expert: $L^*$- best row against all the columns played.

- best row against $\sum_t Ay_t$ and $Ty^* = \sum_t y_t$
  $\rightarrow$ best row against $T Ay^*$.
  $\rightarrow L^* \leq T \times R(y^*)$.

Multiplicative Weights: $L \leq (1 + \varepsilon) L^* + \frac{\ln n}{\varepsilon}$

$TC(x^*) \leq (1 + \varepsilon) TR(y^*) + \frac{\ln n}{\varepsilon} \rightarrow C(x^*) \leq (1 + \varepsilon) R(y^*) + \frac{\ln n}{\varepsilon T}$

$\rightarrow C(x^*) - R(y^*) \leq \varepsilon R(y^*) + \frac{\ln n}{\varepsilon T}$.

$T = \frac{\ln n}{\varepsilon^2}$, $R(y^*) \leq 1 \rightarrow C(x^*) - R(y^*) \leq 2\varepsilon$. 
For any $\varepsilon$, there exists an $\varepsilon$-Approximate Equilibrium.
Comments

For any $\varepsilon$, there exists an $\varepsilon$-Approximate Equilibrium.
Does an equilibrium exist?
Comments

For any \( \varepsilon \), there exists an \( \varepsilon \)-Approximate Equilibrium.
Does an equilibrium exist? Yes.
For any $\varepsilon$, there exists an $\varepsilon$-Approximate Equilibrium.

Does an equilibrium exist? Yes.

Something about math here?
For any $\varepsilon$, there exists an $\varepsilon$-Approximate Equilibrium.

Does an equilibrium exist? Yes.

Something about math here? Fixed point theorem.
For any $\varepsilon$, there exists an $\varepsilon$-Approximate Equilibrium.

Does an equilibrium exist? Yes.

Something about math here? Fixed point theorem.

Later: will use geometry, linear programming.

$T = \ln n \varepsilon^2 \rightarrow O(n \log n \varepsilon^2)$. Basically linear!

Versus Linear Programming: $O(n^3 m)$ Basically quadratic.

(Faster linear programming: $O(\sqrt{n} + m)$ linear solution solves.)

Still much slower... and more complicated.

Dynamics: best response, update weight, best response. Also works with both using multiplicative weights.

“In practice.”
For any $\epsilon$, there exists an $\epsilon$-Approximate Equilibrium.

Does an equilibrium exist? Yes.

Something about math here? Fixed point theorem.

Later: will use geometry, linear programming.

Complexity?
For any $\varepsilon$, there exists an $\varepsilon$-Approximate Equilibrium.

Does an equilibrium exist? Yes.

Something about math here? Fixed point theorem.

Later: will use geometry, linear programming.

Complexity?

$$T = \frac{\ln n}{\varepsilon^2}$$
Comments

For any $\varepsilon$, there exists an $\varepsilon$-Approximate Equilibrium.

Does an equilibrium exist? Yes.

Something about math here? Fixed point theorem.

Later: will use geometry, linear programming.

Complexity?

$$T = \frac{\ln n}{\varepsilon^2} \to O(nm^{\frac{\log n}{\varepsilon^2}}).$$
Comments

For any $\epsilon$, there exists an $\epsilon$-Approximate Equilibrium.
Does an equilibrium exist? Yes.
Something about math here? Fixed point theorem.
Later: will use geometry, linear programming.
Complexity?
$$T = \frac{\ln n}{\epsilon^2} \rightarrow O(nm\frac{\log n}{\epsilon^2}).$$ Basically linear!
For any $\varepsilon$, there exists an $\varepsilon$-Approximate Equilibrium.

Does an equilibrium exist? Yes.

Something about math here? Fixed point theorem.

Later: will use geometry, linear programming.

Complexity?

$$T = \frac{\ln n}{\varepsilon^2} \to O(nm \frac{\log n}{\varepsilon^2}).$$

Basically linear!

Versus Linear Programming: $O(n^3 m)$
For any $\epsilon$, there exists an $\epsilon$-Approximate Equilibrium. Does an equilibrium exist? Yes.

Something about math here? Fixed point theorem.

Later: will use geometry, linear programming.

Complexity?

$$T = \frac{\ln n}{\epsilon^2} \rightarrow O(nm \frac{\log n}{\epsilon^2}).$$

Basically linear!

Versus Linear Programming: $O(n^3m)$ Basically quadratic.
For any $\varepsilon$, there exists an $\varepsilon$-Approximate Equilibrium.

Does an equilibrium exist? Yes.

Something about math here? Fixed point theorem.

Later: will use geometry, linear programming.

Complexity?

$$T = \frac{\ln n}{\varepsilon^2} \rightarrow O(nm\frac{\log n}{\varepsilon^2}).$$  Basically linear!

Versus Linear Programming: $O(n^3m)$ Basically quadratic.

(Faster linear programming: $O(\sqrt{n+m})$ linear solution solves.)
Comments

For any $\varepsilon$, there exists an $\varepsilon$-Approximate Equilibrium.

Does an equilibrium exist? Yes.

Something about math here? Fixed point theorem.

Later: will use geometry, linear programming.

Complexity?

$$T = \frac{\ln n}{\varepsilon^2} \to O(nm\frac{\log n}{\varepsilon^2}).$$ Basically linear!

Versus Linear Programming: $O(n^3m)$ Basically quadratic.

(Faster linear programming: $O(\sqrt{n+m})$ linear solution solves.)

Still much slower
For any $\varepsilon$, there exists an $\varepsilon$-Approximate Equilibrium.

Does an equilibrium exist? Yes.

Something about math here? Fixed point theorem.

Later: will use geometry, linear programming.

Complexity?

$$T = \frac{\ln n}{\varepsilon^2} \rightarrow O(nm\frac{\log n}{\varepsilon^2}).$$

Basically linear!

Versus Linear Programming: $O(n^3m)$ Basically quadratic.

(Faster linear programming: $O(\sqrt{n+m})$ linear solution solves.)

Still much slower ... and more complicated.
For any $\varepsilon$, there exists an $\varepsilon$-Approximate Equilibrium.

Does an equilibrium exist? Yes.

Something about math here? Fixed point theorem.

Later: will use geometry, linear programming.

Complexity?

$$T = \frac{\ln n}{\varepsilon^2} \to O(nm\frac{\log n}{\varepsilon^2}) \text{. Basically linear!}$$

Versus Linear Programming: $O(n^3m)$ Basically quadratic.
(Faster linear programming: $O(\sqrt{n+m})$ linear solution solves.)

Still much slower ... and more complicated.

For any $\varepsilon$, there exists an $\varepsilon$-Approximate Equilibrium.

Does an equilibrium exist? Yes.

Something about math here? Fixed point theorem.

Later: will use geometry, linear programming.

Complexity?

$$T = \frac{\ln n}{\varepsilon^2} \to O(nm\frac{\log n}{\varepsilon^2}).$$

Basically linear!

Versus Linear Programming: $O(n^3m)$ Basically quadratic.

(Faster linear programming: $O(\sqrt{n+m})$ linear solution solves.)

Still much slower ... and more complicated.


Also works with both using multiplicative weights.
Comments

For any $\varepsilon$, there exists an $\varepsilon$-Approximate Equilibrium.

Does an equilibrium exist? Yes.

Something about math here? Fixed point theorem.

Later: will use geometry, linear programming.

Complexity?

$$T = \frac{\ln n}{\varepsilon^2} \rightarrow O(nm\frac{\log n}{\varepsilon^2}).$$ Basically linear!

Versus Linear Programming: $O(n^3 m)$ Basically quadratic.

(Faster linear programming: $O(\sqrt{n+m})$ linear solution solves.)

Still much slower ... and more complicated.


Also works with both using multiplicative weights.

“In practice.”
Boosting...
Learning

Learning just a bit.
Learning just a bit.

Example: set of labelled points, find hyperplane that separates.
Learning just a bit.

Example: set of labelled points, find hyperplane that separates.

\[
\begin{array}{ccc}
- & + & + \\
- & + & + \\
+ & - & - \\
- & + & - \\
\end{array}
\]

Looks hard.
Learning just a bit.

Example: set of labelled points, find hyperplane that separates.

\[
\begin{array}{ccc}
- & + & + \\
- & + & + \\
+ & - & - \\
- & + & - \\
+ & - & - \\
\end{array}
\]

Looks hard.

1/2 of them?
Learning just a bit.

Example: set of labelled points, find hyperplane that separates.

\[
\begin{array}{cccc}
- & + & + & + \\
- & + & + & + \\
+ & - & - & - \\
- & + & - & - \\
\end{array}
\]

Looks hard.

1/2 of them? Easy.
Learning just a bit.

Example: set of labelled points, find hyperplane that separates.

![Diagram of labelled points]

Looks hard.

1/2 of them? Easy.
Arbitrary line.
Learning just a bit.

Example: set of labelled points, find hyperplane that separates.

![Graph showing labelled points and a hyperplane.]

Looks hard.

1/2 of them? Easy.

Arbitrary line. And Scan.
Learning just a bit.

Example: set of labelled points, find hyperplane that separates.

```
-    +    +    +
-    +    +    +
+    -    -    -
-    +    -    -
```

Looks hard.

1/2 of them? Easy.
Arbitrary line. And Scan.
Learning just a bit.

Example: set of labelled points, find hyperplane that separates.

\[
\begin{array}{c|ccccc}
- & + & + & - \\
- & + & + & - \\
+ & - & + & - \\
- & + & - & - \\
- & + & - & - \\
\end{array}
\]

Looks hard.

1/2 of them? Easy.
Arbitrary line. And Scan.

Useless.
Learning just a bit.

Example: set of labelled points, find hyperplane that separates.

- + +
- + +
+ - -
- + -

Looks hard.

1/2 of them? Easy.
Arbitrary line. And Scan.

Useless. A bit more than 1/2 Correct would be better.
Learning just a bit.

Example: set of labelled points, find hyperplane that separates.

\[
\begin{array}{c|cc}
- & + & + \\
- & + & + \\
+ & - & - \\
- & + & - \\
\end{array}
\]

Looks hard.

1/2 of them? Easy.

Arbitrary line. And Scan.

Useless. A bit more than 1/2 Correct would be better.

Weak Learner: Classify $\geq \frac{1}{2} + \varepsilon$ points correctly.
Learning

Learning just a bit.

Example: set of labelled points, find hyperplane that separates.

\[ - \quad + \quad + \quad + \]
\[ + \quad - \quad - \quad - \]

Looks hard.

1/2 of them? Easy.

Arbitrary line. And Scan.

Useless. A bit more than 1/2 Correct would be better.

Weak Learner: Classify \( \geq \frac{1}{2} + \epsilon \) points correctly.

Not really important but ...
Learning

Learning just a bit.

Example: set of labelled points, find hyperplane that separates.

Looks hard.

1/2 of them? Easy.
Arbitrary line. And Scan.

Useless. A bit more than 1/2 Correct would be better.

Weak Learner: Classify $\geq \frac{1}{2} + \varepsilon$ points correctly.

Not really important but ...
Weak Learner/Strong Learner

Input: $n$ labelled points.
Weak Learner/Strong Learner

Input: $n$ labelled points.

Weak Learner:

That's a really strong learner!

Same thing?
Can one use weak learning to produce strong learner?

Boosting: use a weak learner to produce strong learner.
Input: $n$ labelled points.

Weak Learner:
- produce hypothesis correctly classifies $\frac{1}{2} + \epsilon$ fraction
Weak Learner/Strong Learner

Input: \( n \) labelled points.

Weak Learner:
- produce hypothesis correctly classifies \( \frac{1}{2} + \varepsilon \) fraction

Strong Learner:
Weak Learner/Strong Learner

Input: \( n \) labelled points.

Weak Learner:
produce hypothesis correctly classifies \( \frac{1}{2} + \varepsilon \) fraction

Strong Learner:
produce hyp. correctly classifies \( 1 + \mu \) fraction
Input: $n$ labelled points.

Weak Learner:
- produce hypothesis correctly classifies $\frac{1}{2} + \epsilon$ fraction

Strong Learner:
- produce hyp. correctly classifies $1 + \mu$ fraction
  That’s a really strong learner!
Weak Learner/Strong Learner

Input: $n$ labelled points.

Weak Learner:
produce hypothesis correctly classifies $\frac{1}{2} + \epsilon$ fraction

Strong Learner:
produce hyp. correctly classifies $1 + \mu$ fraction

That's a really strong learner!
produce hypothesis correctly classifies $1 - \mu$ fraction
Input: \( n \) labelled points.

Weak Learner:
produce hypothesis correctly classifies \( \frac{1}{2} + \varepsilon \) fraction

Strong Learner:
produce hyp. correctly classifies \( 1 + \mu \) fraction

That’s a really strong learner!
produce hypothesis correctly classifies \( 1 - \mu \) fraction

Same thing?
Weak Learner/Strong Learner

Input: \( n \) labelled points.

Weak Learner:
produce hypothesis correctly classifies \( \frac{1}{2} + \varepsilon \) fraction

Strong Learner:
produce hyp. correctly classifies \( 1 + \mu \) fraction

That’s a really strong learner!
produce hypothesis correctly classifies \( 1 - \mu \) fraction

Same thing?
Can one use weak learning to produce strong learner?
Weak Learner/Strong Learner

Input: $n$ labelled points.

Weak Learner:
produce hypothesis correctly classifies $\frac{1}{2} + \varepsilon$ fraction

Strong Learner:
produce hyp. correctly classifies $1 + \mu$ fraction

That’s a really strong learner!
produce hypothesis correctly classifies $1 - \mu$ fraction

Same thing?

Can one use weak learning to produce strong learner?

Boosting: use a weak learner to produce strong learner.
Poll.

Given a weak learning method (produce ok hypotheses.)
Poll.

Given a weak learning method (produce ok hypotheses.) produce a great hypothesis.
Given a weak learning method (produce ok hypotheses.) produce a great hypothesis.

Can we do this?
Poll.

Given a weak learning method (produce ok hypotheses.) produce a great hypothesis.

Can we do this?

(A) Yes

(B) No
Poll.

Given a weak learning method (produce ok hypotheses.) produce a great hypothesis.

Can we do this?

(A) Yes
(B) No

If yes.
Given a weak learning method (produce ok hypotheses.)
produce a great hypothesis.

Can we do this?

(A) Yes
(B) No

If yes. How?
Given a weak learning method (produce ok hypotheses.) produce a great hypothesis.

Can we do this?

(A) Yes
(B) No

If yes. How?

Multiplicative Weights!
Poll.

Given a weak learning method (produce ok hypotheses.) produce a great hypothesis.

Can we do this?

(A) Yes
(B) No

If yes. How?

Multiplicative Weights!

The endpoint to a line of research.
Boosting/MW Framework

Experts are points.
Experts are points. “Adversary” weak learner.
Experts are points. “Adversary” weak learner. Points (experts) suffer loss when classified correctly.
Boosting/MW Framework

Experts are points. “Adversary” weak learner.
Points (experts) suffer loss when classified correctly.
Learner (adversary) wants to maximize probability

Claim: $h(x)$ is correct on $1-\mu$ of the points!
Experts are points. “Adversary” weak learner.

Points (experts) suffer loss when classified correctly.

Learner (adversary) wants to maximize probability of classifying random point correctly.
Experts are points. “Adversary” weak learner.
Points (experts) suffer loss when classified correctly.
Learner (adversary) wants to maximize probability of classifying random point correctly.
Strong learner algorithm will come from adversary.

\[ T = 2 \gamma^2 \ln \frac{1}{\mu} \]

1. Row player: multiplicative weights \((1 - \gamma)\) on points.
2. Column: run weak learner on row distribution.
3. Hypothesis \( h(x) \): majority of \( h_1(x), h_2(x), \ldots, h_T(x) \).

Claim: \( h(x) \) is correct on \( 1 - \mu \) of the points!
Experts are points. “Adversary” weak learner.
Points (experts) suffer loss when classified correctly.
Learner (adversary) wants to maximize probability of classifying random point correctly.
Strong learner algorithm will come from adversary.
Do $T = \frac{2}{\gamma^2} \ln \frac{1}{\mu}$ rounds
Boosting/MW Framework

Experts are points. “Adversary” weak learner.

Points (experts) suffer loss when classified correctly.

Learner (adversary) wants to maximize probability of classifying random point correctly.

Strong learner algorithm will come from adversary.

Do $T = \frac{2}{\gamma^2} \ln \frac{1}{\mu}$ rounds

1. Row player: multiplicative weights $(1 - \gamma)$ on points.
Boosting/MW Framework

Experts are points. “Adversary” weak learner.
Points (experts) suffer loss when classified correctly.
Learner (adversary) wants to maximize probability of classifying random point correctly.
Strong learner algorithm will come from adversary.

Do $T = \frac{2}{\gamma^2} \ln \frac{1}{\mu}$ rounds

1. Row player: multiplicative weights $(1 - \gamma)$ on points.
2. Column: run weak learner on row distribution.
Boosting/MW Framework

Experts are points. “Adversary” weak learner.

Points (experts) suffer loss when classified correctly.

Learner (adversary) wants to maximize probability of classifying random point correctly.

Strong learner algorithm will come from adversary.

Do $T = \frac{2}{\gamma^2} \ln \frac{1}{\mu}$ rounds

1. Row player: multiplicative weights ($1 - \gamma$) on points.
2. Column: run weak learner on row distribution.
3. Hypothesis $h(x)$:
Experts are points. “Adversary” weak learner.
Points (experts) suffer loss when classified correctly.
Learner (adversary) wants to maximize probability of classifying random point correctly.
Strong learner algorithm will come from adversary.

Do $T = \frac{2}{\gamma^2} \ln \frac{1}{\mu}$ rounds

1. Row player: multiplicative weights($1 - \gamma$) on points.
2. Column: run weak learner on row distribution.
3. Hypothesis $h(x)$: majority of $h_1(x), h_2(x), \ldots, h_T(x)$. 
Experts are points. “Adversary” weak learner. Points (experts) suffer loss when classified correctly. Learner (adversary) wants to maximize probability of classifying random point correctly. Strong learner algorithm will come from adversary.

Do $T = \frac{2}{\gamma^2} \ln \frac{1}{\mu}$ rounds

1. Row player: multiplicative weights $(1 - \gamma)$ on points.
2. Column: run weak learner on row distribution.
3. Hypothesis $h(x)$: majority of $h_1(x), h_2(x), \ldots, h_T(x)$.

**Claim:** $h(x)$ is correct on $1 - \mu$ of the points
Boosting/MW Framework

Experts are points. “Adversary” weak learner.

Points (experts) suffer loss when classified correctly.

Learner (adversary) wants to maximize probability of classifying random point correctly.

Strong learner algorithm will come from adversary.

Do $T = \frac{2}{\gamma^2} \ln \frac{1}{\mu}$ rounds

1. Row player: multiplicative weights $(1 - \gamma)$ on points.
2. Column: run weak learner on row distribution.
3. Hypothesis $h(x)$: majority of $h_1(x), h_2(x), \ldots, h_T(x)$.

**Claim:** $h(x)$ is correct on $1 - \mu$ of the points!
Experts are points. “Adversary” weak learner.
Points (experts) suffer loss when classified correctly.
Learner (adversary) wants to maximize probability
    of classifying random point correctly.
Strong learner algorithm will come from adversary.
Do $T = \frac{2}{\gamma^2} \ln \frac{1}{\mu}$ rounds
1. Row player: multiplicative weights $(1 - \gamma)$ on points.
2. Column: run weak learner on row distribution.
3. Hypothesis $h(x)$: majority of $h_1(x), h_2(x), \ldots, h_T(x)$.

**Claim:** $h(x)$ is correct on $1 - \mu$ of the points!!
Experts are points. “Adversary” weak learner.
Points (experts) suffer loss when classified correctly.
Learner (adversary) wants to maximize probability of classifying random point correctly.
Strong learner algorithm will come from adversary.
Do \( T = \frac{2}{\gamma^2} \ln \frac{1}{\mu} \) rounds
1. Row player: multiplicative weights\( (1 - \gamma) \) on points.
2. Column: run weak learner on row distribution.
3. Hypothesis \( h(x) \): majority of \( h_1(x), h_2(x), \ldots, h_T(x) \).

**Claim:** \( h(x) \) is correct on \( 1 - \mu \) of the points !!!
Experts are points. “Adversary” weak learner.
Points (experts) suffer loss when classified correctly.
Learner (adversary) wants to maximize probability
of classifying random point correctly.
Strong learner algorithm will come from adversary.
Do $T = \frac{2}{\gamma^2} \ln \frac{1}{\mu}$ rounds
1. Row player: multiplicative weights$(1 - \gamma)$ on points.
2. Column: run weak learner on row distribution.
3. Hypothesis $h(x)$: majority of $h_1(x), h_2(x), \ldots, h_T(x)$.

**Claim:** $h(x)$ is correct on $1 - \mu$ of the points! ! !
Cool!
Experts are points. “Adversary” weak learner.
Points (experts) suffer loss when classified correctly.
Learner (adversary) wants to maximize probability
of classifying random point correctly.
Strong learner algorithm will come from adversary.
Do \( T = \frac{2}{\gamma^2} \ln \frac{1}{\mu} \) rounds
1. Row player: multiplicative weights\((1 - \gamma)\) on points.
2. Column: run weak learner on row distribution.
3. Hypothesis \( h(x) \): majority of \( h_1(x), h_2(x), \ldots, h_T(x) \).

**Claim:** \( h(x) \) is correct on \( 1 - \mu \) of the points !!!
Cool!
Really?
Experts are points. “Adversary” weak learner.

Points (experts) suffer loss when classified correctly.

Learner (adversary) wants to maximize probability of classifying random point correctly.

Strong learner algorithm will come from adversary.

Do $T = \frac{2}{\gamma^2 \ln \frac{1}{\mu}}$ rounds

1. Row player: multiplicative weights $(1 - \gamma)$ on points.
2. Column: run weak learner on row distribution.
3. Hypothesis $h(x)$: majority of $h_1(x), h_2(x), \ldots, h_T(x)$.

**Claim:** $h(x)$ is correct on $1 - \mu$ of the points ! ! !

Cool!

Really? Proof?
Adaboost proof.

**Claim:** \( h(x) \) is correct on \( 1 - \mu \) of the points
Adaboost proof.

**Claim:** $h(x)$ is correct on $1 - \mu$ of the points!
Claim: $h(x)$ is correct on $1 - \mu$ of the points!!
Adaboost proof.

**Claim:** $h(x)$ is correct on $1 - \mu$ of the points !!!
Adaboost proof.

**Claim:** $h(x)$ is correct on $1 - \mu$ of the points ! ! !

Let $S_{bad}$ be the set of points where $h(x)$ is incorrect.
Adaboost proof.

**Claim:** $h(x)$ is correct on $1 - \mu$ of the points ! ! !

Let $S_{bad}$ be the set of points where $h(x)$ is incorrect.

majority of $h_t(x)$ are wrong for $x \in S_{bad}$. 
**Adaboost proof.**

**Claim:** $h(x)$ is correct on $1 - \mu$ of the points ! ! !

Let $S_{bad}$ be the set of points where $h(x)$ is incorrect.

majority of $h_t(x)$ are wrong for $x \in S_{bad}$.

$x \in S_{bad}$ is a good expert
Adaboost proof.

**Claim:** $h(x)$ is correct on $1 - \mu$ of the points ! ! !

Let $S_{bad}$ be the set of points where $h(x)$ is incorrect.

- majority of $h_t(x)$ are wrong for $x \in S_{bad}$.
- $x \in S_{bad}$ is a good expert – loses less than $\frac{1}{2}$ the time.
Adaboost proof.

Claim: $h(x)$ is correct on $1 - \mu$ of the points !!!

Let $S_{bad}$ be the set of points where $h(x)$ is incorrect.

majority of $h_t(x)$ are wrong for $x \in S_{bad}$.

$x \in S_{bad}$ is a good expert – loses less than $\frac{1}{2}$ the time.

$W(T) \geq (1 - \varepsilon)^{\frac{T}{2}} |S_{bad}|$
Adaboost proof.

Claim: $h(x)$ is correct on $1 - \mu$ of the points ! ! !
Let $S_{bad}$ be the set of points where $h(x)$ is incorrect.

majority of $h_t(x)$ are wrong for $x \in S_{bad}$.

$x \in S_{bad}$ is a good expert – loses less than $\frac{1}{2}$ the time.

$$W(T) \geq (1 - \varepsilon) \frac{T}{2} |S_{bad}|$$

Each day, weak learner gets $\geq \frac{1}{2} + \gamma$ payoff.
Adaboost proof.

**Claim:** $h(x)$ is correct on $1 - \mu$ of the points !!!

Let $S_{bad}$ be the set of points where $h(x)$ is incorrect.

- majority of $h_t(x)$ are wrong for $x \in S_{bad}$.
- $x \in S_{bad}$ is a good expert – loses less than $\frac{1}{2}$ the time.

$$W(T) \geq (1 - \varepsilon)^\frac{T}{2} |S_{bad}|$$

Each day, weak learner gets $\geq \frac{1}{2} + \gamma$ payoff.

$\rightarrow L_t \geq \frac{1}{2} + \gamma.$
**Adaboost proof.**

**Claim:** $h(x)$ is correct on $1 - \mu$ of the points ! ! ! 
Let $S_{bad}$ be the set of points where $h(x)$ is incorrect. 

- majority of $h_t(x)$ are wrong for $x \in S_{bad}$. 

$x \in S_{bad}$ is a good expert – loses less than $\frac{1}{2}$ the time.

$$W(T) \geq (1 - \varepsilon)^{\frac{T}{2}}|S_{bad}|$$

Each day, weak learner gets $\geq \frac{1}{2} + \gamma$ payoff.

$\rightarrow L_t \geq \frac{1}{2} + \gamma.$

$\rightarrow$
Claim: \( h(x) \) is correct on \( 1 - \mu \) of the points ! ! !

Let \( S_{bad} \) be the set of points where \( h(x) \) is incorrect.

- majority of \( h_t(x) \) are wrong for \( x \in S_{bad} \).
- \( x \in S_{bad} \) is a good expert – loses less than \( \frac{1}{2} \) the time.

\[
W(T) \geq (1 - \epsilon)^{\frac{T}{2}} |S_{bad}|
\]

Each day, weak learner gets \( \geq \frac{1}{2} + \gamma \) payoff.

\[
\rightarrow L_t \geq \frac{1}{2} + \gamma.
\]

\[
\rightarrow W(T) \leq n(1 - \epsilon)^L
\]
Adaboost proof.

**Claim:** $h(x)$ is correct on $1 - \mu$ of the points !!

Let $S_{bad}$ be the set of points where $h(x)$ is incorrect. The majority of $h_t(x)$ are wrong for $x \in S_{bad}$. $x \in S_{bad}$ is a good expert – loses less than $\frac{1}{2}$ the time.

$$W(T) \geq (1 - \varepsilon)^{T/2} |S_{bad}|$$

Each day, weak learner gets $\geq \frac{1}{2} + \gamma$ payoff.

$\rightarrow L_t \geq \frac{1}{2} + \gamma$.

$\rightarrow W(T) \leq n(1 - \varepsilon)^L \leq ne^{-\varepsilon L}$
Adaboost proof.

**Claim:** $h(x)$ is correct on $1 - \mu$ of the points ! ! !

Let $S_{\text{bad}}$ be the set of points where $h(x)$ is incorrect.

- majority of $h_t(x)$ are wrong for $x \in S_{\text{bad}}$.
- $x \in S_{\text{bad}}$ is a good expert – loses less than $\frac{1}{2}$ the time.

$$W(T) \geq (1 - \varepsilon) \frac{T}{2} |S_{\text{bad}}|$$

Each day, weak learner gets $\geq \frac{1}{2} + \gamma$ payoff.

$\rightarrow L_t \geq \frac{1}{2} + \gamma.$

$\rightarrow W(T) \leq n(1 - \varepsilon)^L \leq ne^{-\varepsilon L} \leq ne^{-\varepsilon(\frac{1}{2} + \gamma)} T$
**Claim:** $h(x)$ is correct on $1 - \mu$ of the points!!!

Let $S_{bad}$ be the set of points where $h(x)$ is incorrect.

- Majority of $h_t(x)$ are wrong for $x \in S_{bad}$.
- $x \in S_{bad}$ is a good expert – loses less than $\frac{1}{2}$ the time.

$$W(T) \geq (1 - \varepsilon) \frac{T}{2} |S_{bad}|$$

Each day, weak learner gets $\geq \frac{1}{2} + \gamma$ payoff.

$$\rightarrow L_t \geq \frac{1}{2} + \gamma.$$ 

$$\rightarrow W(T) \leq n(1 - \varepsilon)^L \leq ne^{-\varepsilon L} \leq ne^{-\varepsilon(\frac{1}{2} + \gamma)} T$$

Combining
**Claim:**  \( h(x) \) is correct on \( 1 - \mu \) of the points!!

Let \( S_{bad} \) be the set of points where \( h(x) \) is incorrect.

majority of \( h_t(x) \) are wrong for \( x \in S_{bad} \).

\( x \in S_{bad} \) is a good expert – loses less than \( \frac{1}{2} \) the time.

\[
W(T) \geq (1 - \varepsilon)^{\frac{T}{2}} |S_{bad}|
\]

Each day, weak learner gets \( \geq \frac{1}{2} + \gamma \) payoff.

\[
\rightarrow L_t \geq \frac{1}{2} + \gamma.
\]

\[
\rightarrow W(T) \leq n(1 - \varepsilon)^L \leq ne^{-\varepsilon L} \leq ne^{-\varepsilon(\frac{1}{2} + \gamma) T}
\]

Combining

\[
|S_{bad}|(1 - \varepsilon)^{T/2} \leq W(T) \leq ne^{-\varepsilon(\frac{1}{2} + \gamma) T}
\]
Calculation..

$$|S_{bad}|(1 - \varepsilon)^{T/2} \leq ne^{-\varepsilon(\frac{1}{2} + \gamma)T}$$
Calculation..

\[ |S_{bad}|(1 - \varepsilon)^{T/2} \leq ne^{-\varepsilon(\frac{1}{2} + \gamma)T} \]

Set \( \varepsilon = \gamma \), take logs.
Calculation..

\[ |S_{bad}|(1 - \varepsilon)^{T/2} \leq n e^{-\varepsilon(\frac{1}{2} + \gamma) T} \]

Set \( \varepsilon = \gamma \), take logs.

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} \ln(1 - \gamma) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \]
Calculation..

\[ |S_{bad}|(1 - \varepsilon)^{T/2} \leq ne^{-\varepsilon(1/2 + \gamma)}T \]

Set \( \varepsilon = \gamma \), take logs.

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} \ln(1 - \gamma) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \]

Again, \(-\gamma - \gamma^2 \leq \ln(1 - \gamma)\),
\[ |S_{bad}|(1 - \varepsilon)^{T/2} \leq n e^{-\varepsilon(1/2 + \gamma)} T \]

Set \( \varepsilon = \gamma \), take logs.

\[
\ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} \ln(1 - \gamma) \leq -\gamma T \left( \frac{1}{2} + \gamma \right)
\]

Again, \( -\gamma - \gamma^2 \leq \ln(1 - \gamma) \),

\[
\ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} (-\gamma - \gamma^2) \leq -\gamma T \left( \frac{1}{2} + \gamma \right)
\]
Calculation..

\[ |S_{bad}|(1 - \varepsilon)^{T/2} \leq n e^{-\varepsilon \left( \frac{1}{2} + \gamma \right) T} \]

Set \( \varepsilon = \gamma \), take logs.

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} \ln(1 - \gamma) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \]

Again, \(-\gamma - \gamma^2 \leq \ln(1 - \gamma)\),

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} (-\gamma - \gamma^2) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq -\frac{\gamma^2 T}{2} \]
Calculation..

$$|S_{bad}|(1-\varepsilon)^{T/2} \leq ne^{-\varepsilon(\frac{1}{2}+\gamma) T}$$

Set $\varepsilon = \gamma$, take logs.

$$\ln\left(\frac{|S_{bad}|}{n}\right) + \frac{T}{2} \ln(1-\gamma) \leq -\gamma T\left(\frac{1}{2} + \gamma\right)$$

Again, $-\gamma - \gamma^2 \leq \ln(1-\gamma)$,

$$\ln\left(\frac{|S_{bad}|}{n}\right) + \frac{T}{2} (-\gamma - \gamma^2) \leq -\gamma T\left(\frac{1}{2} + \gamma\right) \rightarrow \ln\left(\frac{|S_{bad}|}{n}\right) \leq -\frac{\gamma^2 T}{2}$$

And $T = \frac{2}{\gamma^2} \log \mu$, 

The misclassified set is at most $\mu$ fraction of all the points.

The hypothesis correctly classifies $1-\mu$ of the points!
Calculation..

\[ |S_{bad}|(1 - \varepsilon)^{T/2} \leq ne^{-\varepsilon(\frac{1}{2} + \gamma)T} \]

Set \( \varepsilon = \gamma \), take logs.

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} \ln(1 - \gamma) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \]

Again, \(-\gamma - \gamma^2 \leq \ln(1 - \gamma)\),

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} (-\gamma - \gamma^2) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq -\frac{\gamma^2 T}{2} \]

And \( T = \frac{2}{\gamma^2} \log \mu \),

\[ \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq \log \mu \]
Calculation..

\[ |S_{bad}|(1 - \varepsilon)^{T/2} \leq ne^{-\varepsilon\left(\frac{1}{2} + \gamma\right)T} \]

Set \( \varepsilon = \gamma \), take logs.

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} \ln(1 - \gamma) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \]

Again, \(-\gamma - \gamma^2 \leq \ln(1 - \gamma)\),

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} (-\gamma - \gamma^2) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq -\frac{\gamma^2 T}{2} \]

And \( T = \frac{2}{\gamma^2} \log \mu \),

\[ \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq \log \mu \rightarrow \frac{|S_{bad}|}{n} \leq \mu. \]
\[|S_{bad}|(1 - \varepsilon)^{T/2} \leq n e^{-\varepsilon(\frac{1}{2} + \gamma)T} \]

Set \( \varepsilon = \gamma \), take logs.

\[\ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} \ln(1 - \gamma) \leq -\gamma T \left( \frac{1}{2} + \gamma \right)\]

Again, \( -\gamma - \gamma^2 \leq \ln(1 - \gamma)\),

\[\ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} (-\gamma - \gamma^2) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq -\frac{\gamma^2 T}{2}\]

And \( T = \frac{2}{\gamma^2} \log \mu \),

\[\rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq \log \mu \rightarrow \frac{|S_{bad}|}{n} \leq \mu.\]

The misclassified set is at most \( \mu \) fraction of all the points.
\[ |S_{bad}|(1 - \varepsilon)^{T/2} \leq ne^{-\varepsilon(\frac{1}{2} + \gamma)} T \]

Set \( \varepsilon = \gamma \), take logs.

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} \ln(1 - \gamma) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \]

Again, \(-\gamma - \gamma^2 \leq \ln(1 - \gamma)\),

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} (-\gamma - \gamma^2) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq -\frac{\gamma^2 T}{2} \]

And \( T = \frac{2}{\gamma^2} \log \mu \),

\[ \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq \log \mu \rightarrow \frac{|S_{bad}|}{n} \leq \mu. \]

The misclassified set is at most \( \mu \) fraction of all the points.

The hypothesis correctly classifies \( 1 - \mu \) of the points.
Calculation..

\[ |S_{bad}|(1 - \varepsilon)^{T/2} \leq n e^{-\varepsilon(\frac{1}{2} + \gamma)} T \]

Set \( \varepsilon = \gamma \), take logs.

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} \ln (1 - \gamma) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \]

Again, \(-\gamma - \gamma^2 \leq \ln (1 - \gamma)\),

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} (-\gamma - \gamma^2) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq -\frac{\gamma^2 T}{2} \]

And \( T = \frac{2}{\gamma^2} \log \mu \),

\[ \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq \log \mu \rightarrow \frac{|S_{bad}|}{n} \leq \mu. \]

The misclassified set is at most \( \mu \) fraction of all the points.

The hypothesis correctly classifies \( 1 - \mu \) of the points!
Calculation..

\[ |S_{bad}|(1 - \varepsilon)^{T/2} \leq n e^{-\varepsilon(\frac{1}{2} + \gamma)} T \]

Set \( \varepsilon = \gamma \), take logs.

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} \ln(1 - \gamma) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \]

Again, \(-\gamma - \gamma^2 \leq \ln(1 - \gamma)\),

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} (-\gamma - \gamma^2) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq -\frac{\gamma^2 T}{2} \]

And \( T = \frac{2}{\gamma^2} \log \mu \),

\[ \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq \log \mu \rightarrow \frac{|S_{bad}|}{n} \leq \mu. \]

The misclassified set is at most \( \mu \) fraction of all the points.

The hypothesis correctly classifies \( 1 - \mu \) of the points !!
Calculation..

\[ |S_{bad}| (1 - \varepsilon)^{T/2} \leq ne^{-\varepsilon(\frac{1}{2} + \gamma)T} \]

Set \( \varepsilon = \gamma \), take logs.

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} \ln(1 - \gamma) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \]

Again, \( -\gamma - \gamma^2 \leq \ln(1 - \gamma) \),

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} (-\gamma - \gamma^2) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq -\frac{\gamma^2 T}{2} \]

And \( T = \frac{2}{\gamma^2} \log \mu \),

\[ \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq \log \mu \rightarrow \frac{|S_{bad}|}{n} \leq \mu. \]

The misclassified set is at most \( \mu \) fraction of all the points. The hypothesis correctly classifies \( 1 - \mu \) of the points ! ! !
\[ |S_{bad}|(1 - \varepsilon)^{T/2} \leq ne^{-\varepsilon(\frac{1}{2} + \gamma)} T \]

Set \( \varepsilon = \gamma \), take logs.

\[ \ln\left(\frac{|S_{bad}|}{n}\right) + \frac{T}{2} \ln(1 - \gamma) \leq -\gamma T\left(\frac{1}{2} + \gamma\right) \]

Again, \(-\gamma - \gamma^2 \leq \ln(1 - \gamma)\),

\[ \ln\left(\frac{|S_{bad}|}{n}\right) + \frac{T}{2}(-\gamma - \gamma^2) \leq -\gamma T\left(\frac{1}{2} + \gamma\right) \rightarrow \ln\left(\frac{|S_{bad}|}{n}\right) \leq -\frac{\gamma^2 T}{2} \]

And \( T = \frac{2}{\gamma^2} \log \mu \),

\[ \rightarrow \ln\left(\frac{|S_{bad}|}{n}\right) \leq \log \mu \rightarrow \frac{|S_{bad}|}{n} \leq \mu. \]

The misclassified set is at most \( \mu \) fraction of all the points.

The hypothesis correctly classifies \( 1 - \mu \) of the points ! ! !

**Claim:** Multiplicative weights: \( h(x) \) is correct on \( 1 - \mu \) of the points
Calculation..

\[ |S_{bad}|(1 - \varepsilon)^{T/2} \leq ne^{-\varepsilon(\frac{1}{2} + \gamma)T} \]

Set \( \varepsilon = \gamma \), take logs.

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} \ln(1 - \gamma) \leq -\gamma T\left(\frac{1}{2} + \gamma\right) \]

Again, \( -\gamma - \gamma^2 \leq \ln(1 - \gamma) \),

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2}(-\gamma - \gamma^2) \leq -\gamma T\left(\frac{1}{2} + \gamma\right) \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq -\frac{\gamma^2 T}{2} \]

And \( T = \frac{2}{\gamma^2} \log \mu \),

\[ \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq \log \mu \rightarrow \frac{|S_{bad}|}{n} \leq \mu . \]

The misclassified set is at most \( \mu \) fraction of all the points.

The hypothesis correctly classifies \( 1 - \mu \) of the points! ! !

Claim: Multiplicative weights: \( h(x) \) is correct on \( 1 - \mu \) of the points!
Calculation..

\[ |S_{bad}|(1 - \varepsilon)^{T/2} \leq ne^{-\varepsilon(\frac{1}{2} + \gamma)T} \]

Set \( \varepsilon = \gamma \), take logs.

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} \ln(1 - \gamma) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \]

Again, \(-\gamma - \gamma^2 \leq \ln(1 - \gamma)\),

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} (-\gamma - \gamma^2) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq -\frac{\gamma^2 T}{2} \]

And \( T = \frac{2}{\gamma^2} \log \mu \),

\[ \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq \log \mu \rightarrow \frac{|S_{bad}|}{n} \leq \mu. \]

The misclassified set is at most \( \mu \) fraction of all the points.

The hypothesis correctly classifies \( 1 - \mu \) of the points ! ! !

Claim: Multiplicative weights: \( h(x) \) is correct on \( 1 - \mu \) of the points ! ! !
\[ |S_{bad}| (1 - \varepsilon)^{T/2} \leq ne^{-\varepsilon(\frac{1}{2} + \gamma)} T \]

Set \( \varepsilon = \gamma \), take logs.

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} \ln(1 - \gamma) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \]

Again, \(-\gamma - \gamma^2 \leq \ln(1 - \gamma)\),

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} (-\gamma - \gamma^2) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq -\frac{\gamma^2 T}{2} \]

And \( T = \frac{2}{\gamma^2} \log \mu \),

\[ \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq \log \mu \rightarrow \frac{|S_{bad}|}{n} \leq \mu. \]

The misclassified set is at most \( \mu \) fraction of all the points.

The hypothesis correctly classifies \( 1 - \mu \) of the points ! ! !

**Claim:** Multiplicative weights: \( h(x) \) is correct on \( 1 - \mu \) of the points ! ! !
\[ |S_{bad}|(1 - \varepsilon)^{T/2} \leq ne^{-\varepsilon(\frac{1}{2} + \gamma) T} \]

Set \( \varepsilon = \gamma \), take logs.

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} \ln(1 - \gamma) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \]

Again, \(-\gamma - \gamma^2 \leq \ln(1 - \gamma)\),

\[ \ln \left( \frac{|S_{bad}|}{n} \right) + \frac{T}{2} (-\gamma - \gamma^2) \leq -\gamma T \left( \frac{1}{2} + \gamma \right) \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq -\frac{\gamma^2 T}{2} \]

And \( T = \frac{2}{\gamma^2} \log \mu \),

\[ \rightarrow \ln \left( \frac{|S_{bad}|}{n} \right) \leq \log \mu \rightarrow \frac{|S_{bad}|}{n} \leq \mu. \]

The misclassified set is at most \( \mu \) fraction of all the points.

The hypothesis correctly classifies \( 1 - \mu \) of the points !!

**Claim:** Multiplicative weights: \( h(x) \) is correct on \( 1 - \mu \) of the points !!
Weak learner learns over distributions of points not points.
Weak learner learns over distributions of points not points. Make copies of points to simulate distributions.
Some details...

Weak learner learns over distributions of points not points.

Make copies of points to simulate distributions.

Used often in machine learning.
Weak learner learns over distributions of points not points.

Make copies of points to simulate distributions.

Used often in machine learning.
Blending learning methods.
Toll/Congestion

Given: $G = (V, E)$.
Given $(s_1, t_1) \ldots (s_k, t_k)$.
Row: choose routing of all paths.
Column: choose edge.
Row pays if column chooses edge on any path.
Toll/Congestion

Given: $G = (V, E)$.
Given $(s_1, t_1) \ldots (s_k, t_k)$.
Row: choose routing of all paths.
Column: choose edge.
Row pays if column chooses edge on any path.

Matrix:
row for each routing: $r$
Toll/Congestion

Given: $G = (V, E)$.
Given $(s_1, t_1), \ldots, (s_k, t_k)$.
Row: choose routing of all paths.
Column: choose edge.
Row pays if column chooses edge on any path.

Matrix:
row for each routing: $r$
column for each edge: $e$
Toll/Congestion

Given: \( G = (V, E) \).

Given \((s_1, t_1) \ldots (s_k, t_k)\).

Row: choose routing of all paths.

Column: choose edge.

Row pays if column chooses edge on any path.

Matrix:

row for each routing: \( r \)

column for each edge: \( e \)

\( A[r, e] \) is congestion on edge \( e \) by routing \( r \)
Given: \( G = (V, E) \).
Given \((s_1, t_1) \ldots (s_k, t_k)\).
Row: choose routing of all paths.
Column: choose edge.
Row pays if column chooses edge on any path.

Matrix:
row for each routing: \( r \)
column for each edge: \( e \)

\[ A[r, e] \] is congestion on edge \( e \) by routing \( r \)

**Offense: (Best Response.)**
Toll/Congestion

Given: $G = (V, E)$.
Given $(s_1, t_1) \ldots (s_k, t_k)$.
Row: choose routing of all paths.
Column: choose edge.
Row pays if column chooses edge on any path.

Matrix:
row for each routing: $r$
column for each edge: $e$

$A[r, e]$ is congestion on edge $e$ by routing $r$

**Offense: (Best Response.)**
Router: route along shortest paths.
Given: \( G = (V, E) \).
Given \((s_1, t_1) \ldots (s_k, t_k)\).
Row: choose routing of all paths.
Column: choose edge.
Row pays if column chooses edge on any path.

Matrix:
row for each routing: \( r \)
column for each edge: \( e \)

\( A[r, e] \) is congestion on edge \( e \) by routing \( r \)

**Offense: (Best Response.)**
Router: route along shortest paths.
Toll: charge most loaded edge.
Given: \( G = (V, E) \).
Given \((s_1, t_1) \ldots (s_k, t_k)\).
Row: choose routing of all paths.
Column: choose edge.
Row pays if column chooses edge on any path.

Matrix:
row for each routing: \( r \)
column for each edge: \( e \)

\[ A[r, e] \text{ is congestion on edge } e \text{ by routing } r \]

**Offense: (Best Response.)**
*Router: route along shortest paths.*
*Toll: charge most loaded edge.*

**Defense:** Toll: maximize shortest path under tolls.
Toll/Congestion

Given: $G = (V, E)$.
Given $(s_1, t_1) \ldots (s_k, t_k)$.
Row: choose routing of all paths.
Column: choose edge.
Row pays if column chooses edge on any path.

Matrix:
row for each routing: $r$
column for each edge: $e$

$A[r, e]$ is congestion on edge $e$ by routing $r$

**Offense: (Best Response.)**
Router: route along shortest paths.
Toll: charge most loaded edge.

**Defense:** Toll: maximize shortest path under tolls.
Route: minimize max congestion on any edge.
Given: $G = (V, E)$.
Given $(s_1, t_1) \ldots (s_k, t_k)$.
Row: choose routing of all paths.
Column: choose edge.
Row pays if column chooses edge on any path.

Matrix:
row for each routing: $r$
column for each edge: $e$

$A[r, e]$ is congestion on edge $e$ by routing $r$

**Offense: (Best Response.)**
Router: route along shortest paths.
Toll: charge most loaded edge.

**Defense:** Toll: maximize shortest path under tolls.
Route: minimize max congestion on any edge.
Two person game.

Row is router.

An exponential number of rows!

Two person game with experts won't be so easy to implement.

Version with row and column flipped may work.

$A_{e,r}$ - congestion of edge $e$ on routing $r$.

Exponential number of columns.

Multiplicative Weights only maintains $m$ weights.

Adversary only needs to provide best column each day.

Runtime only dependent on $m$ and $T$ (number of days.)
Two person game.

Row is router.

An exponential number of rows!
Two person game.

Row is router.

An exponential number of rows!

Two person game with experts won’t be so easy to implement.
Two person game.

Row is router.
An exponential number of rows!
Two person game with experts won’t be so easy to implement.
Version with row and column flipped may work.
Two person game.

Row is router.
An exponential number of rows!
Two person game with experts won’t be so easy to implement.
Version with row and column flipped may work.
$A[e, r]$ - congestion of edge $e$ on routing $r$. 
Two person game.

Row is router.

An exponential number of rows!

Two person game with experts won’t be so easy to implement.

Version with row and column flipped may work.

$A[e, r]$ - congestion of edge $e$ on routing $r$.

$m$ rows.
Two person game.

Row is router.
An exponential number of rows!
Two person game with experts won’t be so easy to implement.
Version with row and column flipped may work.
\( A[e, r] \) - congestion of edge \( e \) on routing \( r \).
m rows. Exponential number of columns.
Two person game.

Row is router.
An exponential number of rows!
Two person game with experts won’t be so easy to implement.
Version with row and column flipped may work.
$A[e, r]$ - congestion of edge $e$ on routing $r$.
$m$ rows. Exponential number of columns.
Multiplicative Weights only maintains $m$ weights.
Two person game.

Row is router.
An exponential number of rows!
Two person game with experts won’t be so easy to implement.
Version with row and column flipped may work.
$A[e,r]$ - congestion of edge $e$ on routing $r$.
$m$ rows. Exponential number of columns.
Multiplicative Weights only maintains $m$ weights.
Adversary only needs to provide best column each day.
Two person game.

Row is router.
An exponential number of rows!
Two person game with experts won’t be so easy to implement.
Version with row and column flipped may work.
$A[e, r]$ - congestion of edge $e$ on routing $r$.
$m$ rows. Exponential number of columns.
Multiplicative Weights only maintains $m$ weights.
Adversary only needs to provide best column each day.
Runtime only dependent on $m$ and $T$ (number of days.)
Congestion minimization and Experts.

Will use gain and $[0, \rho]$ version of experts:
Congestion minimization and Experts.

Will use gain and $[0, \rho]$ version of experts:

$$G \geq (1 - \epsilon) G^* - \frac{\rho \log n}{\epsilon}. $$
Congestion minimization and Experts.

Will use gain and $[0, \rho]$ version of experts:

$$G \geq (1 - \varepsilon)G^* - \frac{\rho \log n}{\varepsilon}.$$ 

Let $T = \frac{k \log n}{\varepsilon^2}$
Congestion minimization and Experts.

Will use gain and $[0, \rho]$ version of experts:

$$G \geq (1 - \varepsilon) G^* - \frac{\rho \log n}{\varepsilon}.$$  

Let $T = \frac{k \log n}{\varepsilon^2}$

1. Row player runs multiplicative weights on edges:
Congestion minimization and Experts.

Will use gain and $[0, \rho]$ version of experts:

$$G \geq (1 - \varepsilon)G^* - \frac{\rho \log n}{\varepsilon}.$$

Let $T = \frac{k \log n}{\varepsilon^2}$

1. Row player runs multiplicative weights on edges:
   $$w_i = w_i(1 + \varepsilon)\frac{g_i}{k}.$$
Congestion minimization and Experts.

Will use gain and $[0, \rho]$ version of experts:

$$G \geq (1 - \varepsilon)G^* - \frac{\rho \log n}{\varepsilon}.$$

Let $T = \frac{k \log n}{\varepsilon^2}$

1. Row player runs multiplicative weights on edges:

$$w_i = w_i(1 + \varepsilon)^{g_i/k}.$$

2. Column routes all paths along shortest paths.

Claim: The congestion, $c_{\text{max}}$, is at most $C^* + 2k\varepsilon$.

Proof:

$$G \geq G^* - \frac{\rho \log n}{\varepsilon} \rightarrow G \geq G^* - G \leq \varepsilon G^* + \frac{k \log n}{\varepsilon}.$$ 

$G^* \leq T \times C^*$ - each day, gain is avg. congestion $\leq$ opt congestion.

$T = \frac{k \log n}{\varepsilon^2} \rightarrow Tc_{\text{max}} - TC \leq \varepsilon TC^* + \frac{k \log n}{\varepsilon} \rightarrow c_{\text{max}} - C^* \leq \varepsilon C^* + \varepsilon$. 
Congestion minimization and Experts.

Will use gain and \([0, \rho]\) version of experts:

\[
G \geq (1 - \epsilon) G^* - \frac{\rho \log n}{\epsilon}.
\]

Let \( T = \frac{k \log n}{\epsilon^2} \)

1. Row player runs multiplicative weights on edges:
   \[
   w_i = w_i (1 + \epsilon) \frac{g_i}{k}.
   \]

2. Column routes all paths along shortest paths.

3. Output the average of all routings: \( \frac{1}{T} \sum_t f(t) \).
Congestion minimization and Experts.

Will use gain and $[0, \rho]$ version of experts:

$$G \geq (1 - \varepsilon)G^* - \frac{\rho \log n}{\varepsilon}.$$  

Let $T = \frac{k \log n}{\varepsilon^2}$

1. Row player runs multiplicative weights on edges:
   $$w_i = w_i (1 + \varepsilon) \frac{g_i}{k}.$$  

2. Column routes all paths along shortest paths.

3. Output the average of all routings:
   $$\frac{1}{T} \sum_t f(t).$$

Claim: The congestion, $c_{\text{max}}$ is at most $C^* + 2k\varepsilon$. 
Congestion minimization and Experts.

Will use gain and $[0, \rho]$ version of experts:

$$G \geq (1 - \varepsilon)G^* - \frac{\rho \log n}{\varepsilon}.$$ 

Let $T = \frac{k \log n}{\varepsilon^2}$

1. Row player runs multiplicative weights on edges:
   $$w_i = w_i(1 + \varepsilon)g_i/k.$$ 
2. Column routes all paths along shortest paths.
3. Output the average of all routings: $\frac{1}{T} \sum_t f(t)$.

**Claim:** The congestion, $c_{max}$ is at most $C^* + 2k\varepsilon$.

Proof:
Congestion minimization and Experts.

Will use gain and $[0, \rho]$ version of experts:

$$G \geq (1 - \varepsilon) G^* - \frac{\rho \log n}{\varepsilon}.$$  

Let $T = \frac{k \log n}{\varepsilon^2}$.

1. Row player runs multiplicative weights on edges:
   $$w_i = w_i(1 + \varepsilon)\frac{g_i}{k}.$$  

2. Column routes all paths along shortest paths.

3. Output the average of all routings:
   $$\frac{1}{T} \sum_t f(t).$$

**Claim:** The congestion, $c_{max}$ is at most $C^* + 2k\varepsilon$.

**Proof:**

$$G \geq G^*(1 - \varepsilon) - \frac{k \log n}{\varepsilon T}.$$
Congestion minimization and Experts.

Will use gain and \([0, \rho]\) version of experts:

\[
G \geq (1 - \varepsilon)G^* - \frac{\rho \log n}{\varepsilon}.
\]

Let \(T = \frac{k \log n}{\varepsilon^2}\)

1. Row player runs multiplicative weights on edges:
   \[
   w_i = w_i(1 + \varepsilon)^{g_i/k}.
   \]

2. Column routes all paths along shortest paths.

3. Output the average of all routings: \(\frac{1}{T} \sum_t f(t)\).

**Claim:** The congestion, \(c_{\text{max}}\) is at most \(C^* + 2k\varepsilon\).

**Proof:**

\[
G \geq G^*(1 - \varepsilon) - \frac{k \log n}{\varepsilon T} \rightarrow G^* - G \leq \varepsilon G^* + \frac{k \log n}{\varepsilon}\]
Congestion minimization and Experts.

Will use gain and $[0, \rho]$ version of experts:

$$G \geq (1 - \varepsilon)G^* - \frac{\rho \log n}{\varepsilon}.$$ 

Let $T = \frac{k \log n}{\varepsilon^2}$

1. Row player runs multiplicative weights on edges:
   $$w_i = w_i(1 + \varepsilon)\frac{g_i}{k}.$$

2. Column routes all paths along shortest paths.

3. Output the average of all routings: $\frac{1}{T} \sum_t f(t)$.

Claim: The congestion, $c_{max}$ is at most $C^* + 2k\varepsilon$.

Proof:

$$G \geq G^*(1 - \varepsilon) - \frac{k \log n}{\varepsilon T} \rightarrow G^* - G \leq \varepsilon G^* + \frac{k \log n}{\varepsilon}$$
Congestion minimization and Experts.

Will use gain and \([0, \rho]\) version of experts:
\[
G \geq (1 - \varepsilon)G^* - \frac{\rho \log n}{\varepsilon}.
\]

Let \(T = \frac{k \log n}{\varepsilon^2}\)

1. Row player runs multiplicative weights on edges:
   \(w_i = w_i(1 + \varepsilon)g_i/k\).

2. Column routes all paths along shortest paths.

3. Output the average of all routings: \(\frac{1}{T} \sum_t f(t)\).

**Claim:** The congestion, \(c_{\text{max}}\) is at most \(C^* + 2k\varepsilon\).

**Proof:**
\[
G \geq G^*(1 - \varepsilon) - \frac{k \log n}{\varepsilon T} \rightarrow G^* - G \leq \varepsilon G^* + \frac{k \log n}{\varepsilon}
\]
\[
G^* = T \cdot c_{\text{max}} - \text{Best row payoff against average routing (times } T).\]
Congestion minimization and Experts.

Will use gain and \([0, \rho]\) version of experts:

\[
G \geq (1 - \varepsilon) G^* - \frac{\rho \log n}{\varepsilon}.
\]

Let \( T = \frac{k \log n}{\varepsilon^2} \)

1. Row player runs multiplicative weights on edges:
   \[
   w_i = w_i (1 + \varepsilon) \frac{g_i}{k}.
   \]

2. Column routes all paths along shortest paths.

3. Output the average of all routings: \( \frac{1}{T} \sum_t f(t) \).

**Claim:** The congestion, \( c_{\text{max}} \) is at most \( C^* + 2k\varepsilon \).

**Proof:**

\[
G \geq G^* (1 - \varepsilon) - \frac{k \log n}{\varepsilon^2 T} \rightarrow G^* - G \leq \varepsilon G^* + \frac{k \log n}{\varepsilon}
\]

\( G^* = T \times c_{\text{max}} \) – Best row payoff against average routing (times \( T \)).

\( G \leq T \times C^* \) – each day, gain is avg. congestion \( \leq \) opt congestion.
Congestion minimization and Experts.

Will use gain and \([0, \rho]\) version of experts:

\[
G \geq (1 - \varepsilon)G^* - \frac{\rho \log n}{\varepsilon}.
\]

Let \(T = \frac{k \log n}{\varepsilon^2}\)

1. Row player runs multiplicative weights on edges:
   \(w_i = w_i(1 + \varepsilon)^{g_i/k} \).

2. Column routes all paths along shortest paths.

3. Output the average of all routings:
   \(\frac{1}{T} \sum_t f(t)\).

**Claim:** The congestion, \(c_{max}\) is at most \(C^* + 2k\varepsilon\).

**Proof:**

\[
G \geq G^*(1 - \varepsilon) - \frac{k \log n}{\varepsilon T} \rightarrow G^* - G \leq \varepsilon G^* + \frac{k \log n}{\varepsilon}
\]

\(G^* = T \times c_{max} \) – Best row payoff against average routing (times \(T\)).

\(G \leq T \times C^* \) – each day, gain is avg. congestion \(\leq\) opt congestion.
Congestion minimization and Experts.

Will use gain and $[0, \rho]$ version of experts:

$$G \geq (1 - \varepsilon) G^* - \frac{\rho \log n}{\varepsilon}.$$  

Let $T = \frac{k \log n}{\varepsilon^2}$

1. Row player runs multiplicative weights on edges:

   $$w_i = w_i (1 + \varepsilon) \frac{g_i}{k}.$$  

2. Column routes all paths along shortest paths.

3. Output the average of all routings: $\frac{1}{T} \sum_t f(t)$.

**Claim:** The congestion, $c_{max}$ is at most $C^* + 2k \varepsilon$.

Proof:

$$G \geq G^* (1 - \varepsilon) - \frac{k \log n}{\varepsilon T} \rightarrow G^* - G \leq \varepsilon G^* + \frac{k \log n}{\varepsilon}$$

$G^* = T \times c_{max}$ – Best row payoff against average routing (times $T$).

$G \leq T \times C^*$ – each day, gain is avg. congestion $\leq$ opt congestion.

$$T = \frac{k \log n}{\varepsilon^2} \rightarrow T c_{max} - TC \leq \varepsilon TC^* + \frac{k \log n}{\varepsilon} \rightarrow$$

$$c_{max} - \hat{C}^* \leq \varepsilon C^* + \varepsilon$$
Better setup.

Runtime: $O(km\log n)$ to route in each step (using Dijkstra’s)
Better setup.

Runtime: $O(km \log n)$ to route in each step (using Dijkstra’s)
$O\left(\frac{k \log n}{\varepsilon^2}\right)$ steps
Runtime: $O(km \log n)$ to route in each step (using Dijkstra’s)
$O\left(\frac{k \log n}{\varepsilon^2}\right)$ steps
to get $c_{\text{max}} - C^* < \varepsilon C^*$ (assuming $C^* > 1$) approximation.
Better setup.

Runtime: $O(km\log n)$ to route in each step (using Dijkstra’s)
$O\left(\frac{k\log n}{\varepsilon^2}\right)$ steps
to get $c_{\max} - C^* < \varepsilon C^*$ (assuming $C^* > 1$) approximation.
To get constant $c$ error.
Better setup.

Runtime: $O(km \log n)$ to route in each step (using Dijkstra’s) $O\left(\frac{k \log n}{\epsilon^2}\right)$ steps

to get $c_{\text{max}} - C^* < \epsilon C^*$ (assuming $C^* > 1$) approximation.

To get constant $c$ error.

$\rightarrow O(k^2 m \log n / \epsilon^2)$ to get a constant approximation.
Better setup.

Runtime: $O(km \log n)$ to route in each step (using Dijkstra’s)

$O\left(\frac{k \log n}{\varepsilon^2}\right)$ steps

to get $c_{\text{max}} - C^* < \varepsilon C^*$ (assuming $C^* > 1$) approximation.

To get constant $c$ error.

$\rightarrow O(k^2 m \log n / \varepsilon^2)$ to get a constant approximation.

(Similar to homework 2 bound that you will get.)
Better setup.

Runtime: $O(km \log n)$ to route in each step (using Dijkstra’s) $O(\frac{k \log n}{\epsilon^2})$ steps to get $c_{\text{max}} - C^* < \epsilon C^*$ (assuming $C^* > 1$) approximation.

To get constant $c$ error.

$\rightarrow O(k^2 m \log n / \epsilon^2)$ to get a constant approximation.

(Similar to homework 2 bound that you will get.)

Homework 3: $O(km \log n)$ algorithm
Better setup.

Runtime: $O(km \log n)$ to route in each step (using Dijkstra’s)

$O\left(\frac{k \log n}{\varepsilon^2}\right)$ steps
to get $c_{\text{max}} - C^* < \varepsilon C^*$ (assuming $C^* > 1$) approximation.

To get constant $c$ error.

$\rightarrow O(k^2 m \log n / \varepsilon^2)$ to get a constant approximation.

(Similar to homework 2 bound that you will get.)

Homework 3: $O(km \log n)$ algorithm!
Better setup.

Runtime: $O(km\log n)$ to route in each step (using Dijkstra’s)
$O\left(\frac{k\log n}{\varepsilon^2}\right)$ steps
to get $c_{\text{max}} - C^* < \varepsilon C^*$ (assuming $C^* > 1$) approximation.

To get constant $c$ error.
→ $O(k^2 m\log n/\varepsilon^2)$ to get a constant approximation.
(Similar to homework 2 bound that you will get.)

Homework 3: $O(km\log n)$ algorithm !!
Better setup.

Runtime: $O(km \log n)$ to route in each step (using Dijkstra’s)
$O\left(\frac{k \log n}{\varepsilon^2}\right)$ steps
to get $c_{\text{max}} - C^* < \varepsilon C^*$ (assuming $C^* > 1$) approximation.
To get constant $c$ error.
$\rightarrow O(k^2 m \log n / \varepsilon^2)$ to get a constant approximation.
(Similar to homework 2 bound that you will get.)

Homework 3: $O(km \log n)$ algorithm !!!
Fractional versus Integer.

Did we (approximately) solve path routing?

Yes?
No?
No!

Average of $T$ routings.

We approximately solved fractional routing problem.

No solution to the path routing problem that is $(1 + \epsilon)$ optimal!
Fractional versus Integer.

Did we (approximately) solve path routing?
Yes?

Average of $T$ routings.

We approximately solved fractional routing problem.

No solution to the path routing problem that is $(1 + \varepsilon)$ optimal!

Homework 2. Problem 1.

Decent solution to path routing problem?

For each $s_i, t_i$, choose path $p_i$ uniformly at random from "daily" paths.

"Concentration" (law of large numbers) $c(e)$ edge has expected congestion, $\tilde{c}(e)$, of $c(e)$.

"Concentration" results later.
Fractional versus Integer.

Did we (approximately) solve path routing?  
Yes? No?

Homework 2. Problem 1. 
Decent solution to path routing problem? 
For each $s_i, t_i$, choose path $p_i$ uniformly at random from "daily" paths.

"Concentration" (law of large numbers) $c(e)$ edge has expected congestion, $\tilde{c}(e)$, of $c(e)$. 
Concentration results? 
Later.
Fractional versus Integer.

Did we (approximately) solve path routing?
Yes? No?
No!
Fractional versus Integer.

Did we (approximately) solve path routing?
Yes? No?
No! Average of $T$ routings.
Fractional versus Integer.

Did we (approximately) solve path routing?  
Yes? No?

No! Average of $T$ routings.  
We approximately solved fractional routing problem.
Fractional versus Integer.

Did we (approximately) solve path routing?
Yes? No?

No! Average of $T$ routings.

We approximately solved fractional routing problem.

No solution to the path routing problem that is $(1 + \epsilon)$ optimal!
Fractional versus Integer.

Did we (approximately) solve path routing?
Yes? No?

No! Average of $T$ routings.
   We approximately solved fractional routing problem.

No solution to the path routing problem that is $(1 + \varepsilon)$ optimal!

Homework 2. Problem 1.
Fractional versus Integer.

Did we (approximately) solve path routing? Yes? No?

No! Average of $T$ routings. We approximately solved fractional routing problem.

No solution to the path routing problem that is $(1 + \varepsilon)$ optimal!

Homework 2. Problem 1.

Decent solution to path routing problem?
Did we (approximately) solve path routing? Yes? No?

No! Average of $T$ routings.

We approximately solved fractional routing problem.

No solution to the path routing problem that is $(1 + \varepsilon)$ optimal!

Homework 2. Problem 1.

Decent solution to path routing problem?

For each $s_i, t_i$, choose path $p_i$ uniformly at random from “daily” paths.
Fractional versus Integer.

Did we (approximately) solve path routing? Yes? No?

No! Average of $T$ routings.

We approximately solved fractional routing problem.

No solution to the path routing problem that is $(1 + \varepsilon)$ optimal!

Homework 2. Problem 1.

Decent solution to path routing problem?

For each $s_i, t_i$, choose path $p_i$ uniformly at random from “daily” paths.

Congestion $c(e)$ edge has expected congestion, $\tilde{c}(e)$, of $c(e)$. 
Fractional versus Integer.

Did we (approximately) solve path routing? Yes? No?

No! Average of $T$ routings. We approximately solved fractional routing problem.

No solution to the path routing problem that is $(1 + \varepsilon)$ optimal!

Homework 2. Problem 1.

Decent solution to path routing problem?

For each $s_i, t_i$, choose path $p_i$ uniformly at random from “daily” paths.

Congestion $c(e)$ edge has expected congestion, $\tilde{c}(e)$, of $c(e)$.

“Concentration” (law of large numbers)
Fractional versus Integer.

Did we (approximately) solve path routing?
Yes? No?

No! Average of $T$ routings.

We approximately solved fractional routing problem.

No solution to the path routing problem that is $(1 + \varepsilon)$ optimal!

Homework 2. Problem 1.

Decent solution to path routing problem?

For each $s_i, t_i$, choose path $p_i$ uniformly at random from “daily” paths.

Congestion $c(e)$ edge has expected congestion, $\tilde{c}(e)$, of $c(e)$.

“Concentration” (law of large numbers)
  $c(e)$ is relatively large ($\Omega(\log n)$)
Fractional versus Integer.

Did we (approximately) solve path routing? Yes? No?

No! Average of $T$ routings. We approximately solved fractional routing problem.

No solution to the path routing problem that is $(1 + \varepsilon)$ optimal!

Homework 2. Problem 1.

Decent solution to path routing problem?

For each $s_i, t_i$, choose path $p_i$ uniformly at random from “daily” paths.

Congestion $c(e)$ edge has expected congestion, $\tilde{c}(e)$, of $c(e)$.

“Concentration” (law of large numbers)

$c(e)$ is relatively large ($\Omega(\log n)$)

$\rightarrow \tilde{c}(e) \approx c(e)$. 
Fractional versus Integer.

Did we (approximately) solve path routing?
Yes? No?

No! Average of $T$ routings.
We approximately solved fractional routing problem.

No solution to the path routing problem that is $(1 + \varepsilon)$ optimal!

Homework 2. Problem 1.

Decent solution to path routing problem?

For each $s_i, t_i$, choose path $p_i$ uniformly at random from “daily” paths.

Congestion $c(e)$ edge has expected congestion, $\tilde{c}(e)$, of $c(e)$.

“Concentration” (law of large numbers)

$c(e)$ is relatively large ($\Omega(\log n)$)

$\rightarrow \tilde{c}(e) \approx c(e)$.

Concentration results?
Fractional versus Integer.

Did we (approximately) solve path routing? Yes? No?

No! Average of $T$ routings.

We approximately solved fractional routing problem.

No solution to the path routing problem that is $(1 + \varepsilon)$ optimal!

Homework 2. Problem 1.

Decent solution to path routing problem?

For each $s_i, t_i$, choose path $p_i$ uniformly at random from “daily” paths.

Congestion $c(e)$ edge has expected congestion, $\tilde{c}(e)$, of $c(e)$.

“Concentration” (law of large numbers)

$c(e)$ is relatively large ($\Omega(\log n)$)

$\rightarrow \tilde{c}(e) \approx c(e)$.

Concentration results? later.
Portfolio Management.

Every day, choose one of \( n \) stocks to invest all your money in.
Portfolio Management.

Every day, choose one of $n$ stocks to invest all your money in.

$c_i^t$ - price of stock on day $t$, 

Experts/multiplicative weights: loss/gains are additive.

Loss/Gain is $\log r$.

Total loss is $\sum_t r(t)$ where $r(t)$ is return on day $t$.

MW: Gives bound on expected loss.

$\sum_t \sum_i P(t) \cdot \log r(t)$ where $P(t)$ is MW distribution on day $t$.

$log x + log y \leq \log (x + y) = \Rightarrow \sum_i P(t) \cdot \log r(t) \leq \log \sum_i P(t) \cdot r(t)$.

Thus expected log of the ratio of the algorithm to the best stock is within $O(\sqrt{\log n} \cdot T)$ of the best. ($\log r \leq 1$).
Portfolio Management.

Every day, choose one of $n$ stocks to invest all your money in. 

$c^t_i$ - price of stock on day $t$, and end of day for $t - 1$. 
Portfolio Management.

Every day, choose one of $n$ stocks to invest all your money in.

$c_i^t$ - price of stock on day $t$, and end of day for $t - 1$.

If invest $P$ in stock $i$, on day $t$. 

$log(x) + log(y) \leq log(x^2 + y^2) = \Rightarrow \sum_i P(t) i \log r(t) i \leq \log \sum_i P(t) i r(t) i$. 

Thus expected log of the ratio of the algorithm to the best stock is within $O(\sqrt{\log n} T)$ of the best. ($log r \leq 1$).
Portfolio Management.

Every day, choose one of \( n \) stocks to invest all your money in.

\( c_i^t \) - price of stock on day \( t \), and end of day for \( t - 1 \).

If invest \( P \) in stock \( i \), on day \( t \).

Have \( \frac{c_i(t)}{c_i} \) \( P \) next day
Portfolio Management.

Every day, choose one of $n$ stocks to invest all your money in.

$c^t_i$ - price of stock on day $t$, and end of day for $t-1$.

If invest $P$ in stock $i$, on day $t$.

Have $\frac{c^{(t)}_i}{c_i} P$ next day \( r^{(t)}_i = \frac{c^{(t)}_i}{c_i} \).
Every day, choose one of \( n \) stocks to invest all your money in.

\( c_i^t \) - price of stock on day \( t \), and end of day for \( t - 1 \).

If invest \( P \) in stock \( i \), on day \( t \).

Have \( \frac{c_i^{(t)}}{c_i} \) \( P \) next day \( (r_i^{(t)} = \frac{c_i^{(t)}}{c_i}) \).
Portfolio Management.

Every day, choose one of \( n \) stocks to invest all your money in.

\( c_i^t \) - price of stock on day \( t \), and end of day for \( t - 1 \).

If invest \( P \) in stock \( i \), on day \( t \).

Have \( \frac{c_i^{(t)}}{c_i} P \) next day \( (r_i^{(t)} = \frac{c_i^{(t)}}{c_i}) \).

Experts/multiplicative weights: loss/gains are additive.
Every day, choose one of $n$ stocks to invest all your money in.

$c^t_i$ - price of stock on day $t$, and end of day for $t-1$.

If invest $P$ in stock $i$, on day $t$.

Have $\frac{c_i^{(t)}}{c_i} P$ next day \( r_i^{(t)} = \frac{c_i^{(t)}}{c_i} \).

Experts/multiplicative weights: loss/gains are additive.

Loss/Gain is log $r$.
Every day, choose one of $n$ stocks to invest all your money in.

$c^t_i$ - price of stock on day $t$, and end of day for $t - 1$.

If invest $P$ in stock $i$, on day $t$.
Have $\frac{c^{(t)}_i}{c_i} P$ next day \( (r^{(t)}_i = \frac{c^{(t)}_i}{c_i}) \).

Experts/multiplicative weights: loss/gains are additive.
Loss/Gain is log $r$.
Total loss is $\sum_t r^{(t)}$ where $r^{(t)}$ is return on day $t$. 

Thus expected log of the ratio of the algorithm to the best stock is within $O(\sqrt{\log n T})$ of the best. ($\log r \leq 1$).
Portfolio Management.

Every day, choose one of $n$ stocks to invest all your money in.

$c_i^t$ - price of stock on day $t$, and end of day for $t-1$.

If invest $P$ in stock $i$, on day $t$.

Have $\frac{c_i(t)}{c_i} P$ next day \((r_i(t) = \frac{c_i(t)}{c_i})\).

Experts/multiplicative weights: loss/gains are additive.

Loss/Gain is $\log r$.

Total loss is $\sum_t r^{(t)}$ where $r^{(t)}$ is return on day $t$. 
Portfolio Management.

Every day, choose one of \( n \) stocks to invest all your money in.

\( c_i^t \) - price of stock on day \( t \), and end of day for \( t - 1 \).

If invest \( P \) in stock \( i \), on day \( t \).

Have \( \frac{c_i(t)}{c_i} P \) next day \( \left( r_i(t) = \frac{c_i(t)}{c_i} \right) \).

Experts/multiplicative weights: loss/gains are additive.

Loss/Gain is \( \log r \).

Total loss is \( \sum_t r^{(t)} \) where \( r^{(t)} \) is return on day \( t \).

MW: Gives bound on \textbf{expected} loss.
Portfolio Management.

Every day, choose one of \( n \) stocks to invest all your money in.

\( c^t_i \) - price of stock on day \( t \), and end of day for \( t - 1 \).

If invest \( P \) in stock \( i \), on day \( t \).

Have \( \frac{c^{(t)}_i}{c_i} P \) next day \( \left( r^{(t)}_i = \frac{c^{(t)}_i}{c_i} \right) \)

Experts/multiplicative weights: loss/gains are additive.

Loss/Gain is \( \log r \).

Total loss is \( \sum_t r^{(t)} \) where \( r^{(t)} \) is return on day \( t \).

MW: Gives bound on expected loss.

\[ \sum_t \sum_i P^{(t)}_i \log r^{(t)}_i \]
Portfolio Management.

Every day, choose one of $n$ stocks to invest all your money in.

$c_i^t$ - price of stock on day $t$, and end of day for $t - 1$.

If invest $P$ in stock $i$, on day $t$.

Have $\frac{c_i^{(t)}}{c_i} P$ next day \( (r_i^{(t)} = \frac{c_i^{(t)}}{c_i}) \).

Experts/multiplicative weights: loss/gains are additive.

Loss/Gain is $\log r$.

Total loss is $\sum_t r^{(t)}$ where $r^{(t)}$ is return on day $t$.

MW: Gives bound on expected loss.

$\sum_t \sum_i P_i^{(t)} \log r^{(t)}$ where $P_i^{(t)}$ is MW distribution on day $t$. 

$$\log x + \log y \leq \log (x + y) \Rightarrow \sum_i P_i^{(t)} \log r^{(t)} \leq \log \sum_i P_i^{(t)} r^{(t)}.$$
Every day, choose one of $n$ stocks to invest all your money in. 

$c_i^t$ - price of stock on day $t$, and end of day for $t - 1$. 

If invest $P$ in stock $i$, on day $t$. 

Have $\frac{c_i^{(t)}}{c_i} P$ next day $\quad (r_i^{(t)} = \frac{c_i^{(t)}}{c_i}.)$

Experts/multiplicative weights: loss/gains are additive. 

Loss/Gain is $\log r$. 

Total loss is $\sum_t r^{(t)}$ where $r^{(t)}$ is return on day $t$. 

MW: Gives bound on expected loss. 

$\sum_t \sum_i P_i^{(t)} \log r^{(t)}_i$ where $P_i^{(t)}$ is MW distribution on day $t$. 

$\log x + \log y \leq \log(\frac{x+y}{2})$
Every day, choose one of $n$ stocks to invest all your money in.  

$c^t_i$ - price of stock on day $t$, and end of day for $t-1$.

If invest $P$ in stock $i$, on day $t$.  

Have $\frac{c^{(t)}_i}{c_{i}} P$ next day  \hspace{1cm} (r^{(t)}_i = \frac{c^{(t)}_i}{c_{i}}.)

Experts/multiplicative weights: loss/gains are additive. 

Loss/Gain is $\log r$.

Total loss is $\sum_t r^{(t)}$ where $r^{(t)}$ is return on day $t$.

MW: Gives bound on expected loss. 

$\sum_t \sum_i P^{(t)}_i \log r^{(t)}_i$ where $P^{(t)}_i$ is MW distribution on day $t$.

$$\frac{\log x+\log y}{2} \leq \log(\frac{x+y}{2}) \implies \sum_i P^{(t)}_i \log r^{(t)}_i \leq \log \sum_i P^{(t)}_i r^{(t)}_i.$$
Portfolio Management.

Every day, choose one of \( n \) stocks to invest all your money in.

\( c^t_i \) - price of stock on day \( t \), and end of day for \( t - 1 \).

If invest \( P \) in stock \( i \), on day \( t \).

Have \( \frac{c_i(t)}{c_i} P \) next day \( \left( r_i(t) = \frac{c_i(t)}{c_i} \right) \)

Experts/multiplicative weights: loss/gains are additive.

Loss/Gain is log \( r \).

Total loss is \( \sum_t r(t) \) where \( r(t) \) is return on day \( t \).

MW: Gives bound on expected loss.

\( \sum_t \sum_i P_i(t) \log r(t)_i \) where \( P_i(t) \) is MW distribution on day \( t \).

\[ \log\frac{x+y}{2} \leq \log \left( \frac{x+y}{2} \right) \implies \sum_i P_i(t) \log r_i(t) \leq \log \sum_i P_i(t) r_i(t). \]

Thus expected log of the ratio of the algorithm to the best stock
Every day, choose one of \( n \) stocks to invest all your money in.

\( c_i^t \) - price of stock on day \( t \), and end of day for \( t - 1 \).

If invest \( P \) in stock \( i \), on day \( t \).

Have \( \frac{c_i^{(t)}}{c_i} P \) next day \( \left( r_i^{(t)} = \frac{c_i^{(t)}}{c_i} . \right) \)

Experts/multiplicative weights: loss/gains are additive.

Loss/Gain is \( \log r \).

Total loss is \( \sum_t r^{(t)} \) where \( r^{(t)} \) is return on day \( t \).

MW: Gives bound on **expected** loss.

\[ \sum_t \sum_i P_i^{(t)} \log r^{(t)} \leq \log \sum_i P_i^{(t)} r_i^{(t)} . \]

Thus expected log of the ratio of the algorithm to the best stock
is within \( O\left(\sqrt{\frac{\log n}{T}}\right) \) of the best. \( (\log r \leq 1) \).
See you on Tuesday.