Twenty (Simple) Questions
Joint work with
Yuval Dagan, Ariel Gabizon, Daniel Kane, Shay Moran

Yuval Filmus, 26 April 2021, HUJI CS Colloquium
The Game of 20 Questions
The Game of 20 Questions

Bob

Thinks of a number between 1 to \( n \)
The Game of 20 Questions

Alice

Finds the number by asking Yes/No questions

Bob

Thinks of a number between 1 to $n$
The Game of 20 Questions

Alice
Finds the number by asking Yes/No questions

Bob
Thinks of a number between 1 to $n$

Cooperative!
The Game of 20 Questions

Alice
Finds the number by asking Yes/No questions

Bob
Thinks of a number between 1 to $n$

binary search: $\log n$ questions

Cooperative!
Distributional 20 Questions
Distributional 20 Questions

Bob

Samples a number between 1 to \( n \) according to \( \mu \)
Distributional 20 Questions

Alice

Finds the number by asking Yes/No questions

Bob

Samples a number between 1 to $n$ according to $\mu$
Distributional 20 Questions

**Alice**

Finds the number by asking Yes/No questions

**Bob**

Samples a number between 1 to $n$ according to $\mu$

$\mu$ known to both parties!
Distributional 20 Questions

Alice

Finds the number by asking Yes/No questions

Bob

Samples a number between 1 to \(n\) according to \(\mu\)

\(\mu\) known to both parties!

Huffman’s algorithm: \(H(\mu)+1\) questions on average
Distributional 20 Questions

Alice

Finds the number by asking Yes/No questions

Bob

Samples a number between 1 to $n$ according to $\mu$

$\mu$ known to both parties!

Huffman’s algorithm: $H(\mu) + 1$ questions on average

$H(\mu) = \text{entropy of } \mu = \text{amortized # questions when solving many games in parallel}$
Distributional 20 Questions

Huffman’s algorithm could involve complicated questions:
Distributional 20 Questions

Huffman’s algorithm could involve complicated questions:

Is $x$ one of $2, 3, 5, 7, 11, 13$?
Distributional 20 Questions

Huffman’s algorithm could involve complicated questions:

Is $x$ one of 2, 3, 5, 7, 11, 13?

What can we accomplish using simple questions?
Binary Search Trees

Diagram:
- Root node: 3
  - Left child: <3?
    - Left child: <2?
      - Left child: 1
      - Right child: 2
    - Right child: 3
  - Right child: N
    - Right child: Y

Explanation:
- The tree follows the binary search tree property, where each node's value is greater than all the values in its left subtree and less than all the values in its right subtree.
- The diagram illustrates how to search for a value: start at the root, compare the value to the root, and follow the left or right path based on the comparison.
Binary Search Trees

Gilbert and Moore: optimal BST achieves $H(\mu) + 2$
Gilbert and Moore: optimal BST achieves $H(\mu)+2$
(optimal for distributions concentrated on some $x \in \{2, \ldots, n-1\}$)
Gilbert–Moore Algorithm

Binary search over [0,1]
Gilbert–Moore Algorithm

Binary search over [0,1]
Gilbert–Moore Algorithm

Binary search over [0,1]
Gilbert–Moore Algorithm

Binary search over [0,1]
Gilbert–Moore Algorithm

Binary search over [0,1]
Gilbert–Moore Algorithm

Binary search over [0,1]

Rissanen–Horibe Algorithm

Ask most informative question
Gilbert–Moore Algorithm

Binary search over [0,1]

Rissanen–Horibe Algorithm

Ask most informative question
Gilbert–Moore Algorithm

Binary search over $[0,1]$  

Rissanen–Horibe Algorithm

Ask most informative question
Gilbert–Moore Algorithm

Binary search over [0,1]

Rissanen–Horibe Algorithm

Ask most informative question
Gilbert–Moore Algorithm

Both algorithms guarantee $H(\mu)+2$

Rissanen–Horibe Algorithm

Ask most informative question
Analysis of Gilbert–Moore
Analysis of Gilbert–Moore
Analysis of Gilbert–Moore
Analysis of Gilbert–Moore

After $k$ questions, zero in on interval of length $2^{-k}$
Analysis of Gilbert–Moore

After $k$ questions, zero in on interval of length $2^{-k}$
Can stop once interval has length at most $\mu(x)/2$
After $k$ questions, zero in on interval of length $2^{-k}$
Can stop once interval has length at most $\mu(x)/2$
Stop after $\lceil \log(2/\mu(x)) \rceil < \log(1/\mu(x)) + 2$ questions
Binary Split Trees

<2?

N

Y

N

Y

1

2

3

1

3
We show: optimal binary split tree achieves $H(\mu)+1$
We show: optimal binary split tree achieves $H(\mu)+1$

Same performance guarantee as Huffman!
Our Algorithm
Our Algorithm

If most probable element $i$ has probability $\geq 0.3$:
   Ask if $x = i$
Otherwise:
   Ask most informative “<” question
Our Algorithm

If most probable element \( i \) has probability \( \geq 0.3 \):
   Ask if \( x = i \)
Otherwise:
   Ask most informative “<” question
Our Algorithm

If most probable element $i$ has probability $\geq 0.3$:
   Ask if $x = i$
Otherwise:
   Ask most informative “<” question
Our Algorithm

If most probable element $i$ has probability $\geq 0.3$:
   Ask if $x = i$
Otherwise:
   Ask most informative “<” question

Why do we care?
Chunked Binary Split Trees
### Chunked Binary Split Trees

<table>
<thead>
<tr>
<th>word 0</th>
<th>word 1</th>
<th>word 2</th>
<th>word 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>=AB01?</td>
<td>&lt;BDBB?</td>
<td>=0010?</td>
<td>&lt;0042?</td>
</tr>
<tr>
<td>&lt;C0A1?</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Chunked Binary Split Trees**

<table>
<thead>
<tr>
<th>word 0</th>
<th>word 1</th>
<th>word 2</th>
<th>word 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>=AB01?</td>
<td>&lt;BDBB?</td>
<td>=0010?</td>
<td>&lt;0042?</td>
</tr>
<tr>
<td>&lt;C0A1?</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Performance on $w$ words: $H(\mu) + w$
Chunked Binary Split Trees

\[
\begin{array}{cccc}
\text{word 0} & \text{word 1} & \text{word 2} & \text{word 3} \\
=AB01? & <BDBB? & =0010? & <0042? \\
<C0A1? & \\
\end{array}
\]

Performance on \( w \) words: \( H(\mu) + w \)

Number of different questions: \( 2wn^{1/w} \)
Chunked Binary Split Trees

<table>
<thead>
<tr>
<th>word 0</th>
<th>word 1</th>
<th>word 2</th>
<th>word 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>=AB01?</td>
<td>&lt;BDBB?</td>
<td>=0010?</td>
<td>&lt;0042?</td>
</tr>
<tr>
<td>&lt;C0A1?</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Performance on $w$ words: $H(\mu) + w$

Number of different questions: $2wn^{1/w}$

Optimal for redundancy $w$!
Playing 20 Questions with a Liar
Playing 20 Questions with a Liar

Thinks of a number between 1 to $n$
Playing 20 Questions with a Liar

Alice

Finds the number by asking Yes/No questions

Bob

Thinks of a number between 1 to n
Playing 20 Questions with a Liar

Alice

Finds the number by asking Yes/No questions

Bob

Thinks of a number between 1 to n

Bob allowed to lie k times
Playing 20 Questions with a Liar

Alice

Finds the number by asking Yes/No questions

Bob

Thinks of a number between 1 to $n$

optimal cost: $\log n + k\log\log n$

Bob allowed to lie $k$ times
Distributional 20 Questions with a Liar
Distributional 20 Questions with a Liar

Bob

Samples a number between 1 to $n$ according to $\mu$
Distributional 20 Questions with a Liar

**Alice**
Finds the number by asking Yes/No questions

**Bob**
Samples a number between 1 to \( n \) according to \( \mu \)
Distributional 20 Questions with a Liar

Alice

Finds the number by asking Yes/No questions

Bob

Samples a number between 1 to \( n \) according to \( \mu \)

Bob allowed to lie \( k \) times
Distributional 20 Questions with a Liar

**Alice**
Finds the number by asking Yes/No questions

**Bob**
Samples a number between 1 to \( n \) according to \( \mu \)

Bob allowed to lie \( k \) times

**Optimal cost:**
\[
H(\mu) + kH_2(\mu)
\]
on average
Distributional 20 Questions with a Liar

**Alice**

Finds the number by asking Yes/No questions

**Bob**

Samples a number between 1 to \( n \) according to \( \mu \)

Bob allowed to lie \( k \) times

**Optimal cost:** 
\[ H(\mu) + kH_2(\mu) \] on average

**Formulas:**
- \( H(\mu) = E[\log \frac{1}{\mu}] \)
- \( H_2(\mu) = E[\log \log \frac{1}{\mu}] \)
Gilbert–Moore with Lies
Gilbert–Moore with Lies

\[ \mu(1) \quad \mu(2) \quad \mu(3) \quad \mu(4) \quad \mu(5) \quad \mu(6) \]
Gilbert–Moore with Lies
Gilbert–Moore with Lies

0 \quad \mu(1) \quad \mu(2) \quad \mu(3) \quad 1/2 \quad \mu(4) \quad \mu(5) \quad \mu(6) \quad 1

Lie!
Gilbert–Moore with Lies
Gilbert–Moore with Lies
Gilbert–Moore with Lies

\[ \mu(1) \quad \mu(2) \quad \mu(3) \quad \mu(4) \quad \mu(5) \quad \mu(6) \]
Gilbert–Moore with Lies

After first lie, answer always “>” – suspicious!
Gilbert–Moore with Lies

After first lie, answer always “>” – suspicious!

Figure out true answer, possibly rollback
Why $kH_2(\mu)$ is right overhead?

$H(\mu) = E[\log 1/\mu] \quad H_2(\mu) = E[\log\log 1/\mu]$
Why $kH_2(\mu)$ is right overhead?

\[
H(\mu) = \mathbb{E}[\log 1/\mu] \quad H_2(\mu) = \mathbb{E}[\log\log 1/\mu]
\]

Lower bound:
Why $kH_2(\mu)$ is right overhead?

$$H(\mu) = E[\log 1/\mu] \qquad H_2(\mu) = E[\log\log 1/\mu]$$

Lower bound:

At end of game, Alice knows both $x$ and positions where Bob lied
Why $kH_2(\mu)$ is right overhead?

\[
H(\mu) = E[\log 1/\mu] \quad H_2(\mu) = E[\log\log 1/\mu]
\]

Lower bound:
At end of game, Alice knows both $x$ and positions where Bob lied.
Game lasts for $\approx \log(1/\mu(x))$ rounds.
Why $kH_2(\mu)$ is right overhead?

$H(\mu) = \mathbb{E}[\log 1/\mu] \quad H_2(\mu) = \mathbb{E}[\log\log 1/\mu]$

Lower bound:
At end of game, Alice knows both $x$ and positions where Bob lied
Game lasts for $\approx \log(1/\mu(x))$ rounds
Each lie position requires Alice to find $\log\log(1/\mu(x))$ more bits
Why $kH_2(\mu)$ is right overhead?

\[
H(\mu) = E[\log 1/\mu] \quad H_2(\mu) = E[\log\log 1/\mu]
\]

**Lower bound:**
At end of game, Alice knows both $x$ and positions where Bob lied
Game lasts for $\approx \log(1/\mu(x))$ rounds
Each lie position requires Alice to find $\log\log(1/\mu(x))$ more bits

**Upper bound:**
Why $kH_2(\mu)$ is right overhead?

$$H(\mu) = E[\log 1/\mu] \quad H_2(\mu) = E[\log\log 1/\mu]$$

**Lower bound:**
At end of game, Alice knows both $x$ and positions where Bob lied
Game lasts for $\approx \log(1/\mu(x))$ rounds
Each lie position requires Alice to find $\log\log(1/\mu(x))$ more bits

**Upper bound:**
Length of suspicion interval balances “false positive” and overhead
Why \( kH_2(\mu) \) is right overhead?

\[ H(\mu) = E[\log 1/\mu] \quad H_2(\mu) = E[\log \log 1/\mu] \]

Lower bound:
At end of game, Alice knows both \( x \) and positions where Bob lied
Game lasts for \( \approx \log(1/\mu(x)) \) rounds
Each lie position requires Alice to find \( \log \log(1/\mu(x)) \) more bits

Upper bound:
Length of suspicion interval balances “false positive” and overhead
Optimal choice turns out to be \( \log(\text{depth}) \approx \log \log(1/\mu(x)) \)
Why $kH_2(\mu)$ is right overhead?

\[ H(\mu) = \mathbb{E}[\log \frac{1}{\mu}] \quad H_2(\mu) = \mathbb{E}[\log \log \frac{1}{\mu}] \]

Lower bound:
At end of game, Alice knows both $x$ and positions where Bob lied
Game lasts for $\approx \log(1/\mu(x))$ rounds
Each lie position requires Alice to find $\log \log(1/\mu(x))$ more bits

Upper bound:
Length of suspicion interval balances “false positive” and overhead
Optimal choice turns out to be $\log(\text{depth}) \approx \log \log(1/\mu(x))$
Cost incurred once per lie
Matching Huffman’s algorithm exactly
Matching Huffman’s algorithm exactly

Can we match Huffman exactly using a subset of all possible questions?
Matching Huffman’s algorithm exactly

Can we match Huffman exactly using a subset of all possible questions?
Matching Huffman’s algorithm exactly

Can we match Huffman exactly using a subset of all possible questions?
Matching Huffman’s algorithm exactly

Can we match Huffman exactly using a subset of all possible questions?

Enough to handle “dyadic” distributions
Matching Huffman’s algorithm exactly

Enough to show:
Each dyadic distribution $\mu$ has a strategy using $H(\mu)$ questions in expectation
Matching Huffman’s algorithm exactly

Enough to show:
Each dyadic distribution $\mu$ has a strategy using $H(\mu)$ questions in expectation

Equivalently:
Can always find question splitting $\mu$ evenly
Matching Huffman’s algorithm exactly

Enough to show:
Each dyadic distribution $\mu$ has a strategy
using $H(\mu)$ questions in expectation

Equivalently:
Can always find question splitting $\mu$ evenly

\[
\begin{align*}
1/4 & 1/16 & 1/8 & 1/8 & 1/16 & 1/16 & 1/16 & 1/4 \\
\mu(1) & \mu(2) & \mu(3) & \mu(4) & \mu(5) & \mu(6) & \mu(7) & \mu(8)
\end{align*}
\]
Matching Huffman’s algorithm exactly

Enough to show:
Each dyadic distribution $\mu$ has a strategy using $H(\mu)$ questions in expectation

Equivalently:
Can always find question splitting $\mu$ evenly

$\mu(1) = \frac{1}{4}$
$\mu(2) = \frac{1}{16}$
$\mu(3) = \frac{1}{8}$
$\mu(4) = \frac{1}{8}$
$\mu(5) = \frac{1}{16}$
$\mu(6) = \frac{1}{16}$
$\mu(7) = \frac{1}{16}$
$\mu(8) = \frac{1}{4}$
Matching Huffman’s algorithm exactly

Goal: Can always find question splitting $\mu$ evenly

$\frac{1}{4}$  $\frac{1}{16}$  $\frac{1}{8}$  $\frac{1}{8}$  $\frac{1}{16}$  $\frac{1}{16}$  $\frac{1}{16}$  $\frac{1}{4}$

$\mu(1)$  $\mu(2)$  $\mu(3)$  $\mu(4)$  $\mu(5)$  $\mu(6)$  $\mu(7)$  $\mu(8)$
Matching Huffman’s algorithm exactly

Goal: Can always find question splitting $\mu$ evenly

$\mu(1)$  $\mu(2)$  $\mu(3)$  $\mu(4)$  $\mu(5)$  $\mu(6)$  $\mu(7)$  $\mu(8)$

$\frac{1}{4}$  $\frac{1}{16}$  $\frac{1}{8}$  $\frac{1}{8}$  $\frac{1}{16}$  $\frac{1}{16}$  $\frac{1}{16}$  $\frac{1}{4}$

Example: all subsets of $\{1,\ldots,n/2\}$ + all subsets of $\{n/2+1,\ldots,n\}$
Matching Huffman’s algorithm exactly

Goal: Can always find question splitting \( \mu \) evenly

\[
\begin{align*}
\mu(1) &= 1/4 \\
\mu(2) &= 1/16 \\
\mu(3) &= 1/8 \\
\mu(4) &= 1/8 \\
\mu(5) &= 1/16 \\
\mu(6) &= 1/16 \\
\mu(7) &= 1/16 \\
\mu(8) &= 1/4
\end{align*}
\]

Example: all subsets of \( \{1, \ldots, n/2\} \) + all subsets of \( \{n/2+1, \ldots, n\} \)

Either \( \mu(\{1, \ldots, n/2\}) \geq 1/2 \) or \( \mu(\{n/2+1, \ldots, n\}) \geq 1/2 \), say the former
Matching Huffman’s algorithm exactly

Goal: Can always find question splitting $\mu$ evenly

Example: all subsets of $\{1, \ldots, n/2\} + \text{all subsets of } \{n/2+1, \ldots, n\}$

Either $\mu(\{1, \ldots, n/2\}) \geq 1/2$ or $\mu(\{n/2+1, \ldots, n\}) \geq 1/2$, say the former

Arrange elements in non-increasing order of probability

\begin{align*}
\mu(1) &= 1/4 \\
\mu(2) &= 1/16 \\
\mu(3) &= 1/8 \\
\mu(4) &= 1/8 \\
\mu(5) &= 1/16 \\
\mu(6) &= 1/16 \\
\mu(7) &= 1/16 \\
\mu(8) &= 1/4
\end{align*}
Matching Huffman’s algorithm exactly

Goal: Can always find question splitting $\mu$ evenly

Example: all subsets of $\{1, \ldots, n/2\}$ + all subsets of $\{n/2+1, \ldots, n\}$

Either $\mu(\{1, \ldots, n/2\}) \geq 1/2$ or $\mu(\{n/2+1, \ldots, n\}) \geq 1/2$, say the former

Arrange elements in non-increasing order of probability

Some prefix sums to exactly 1/2
Matching Huffman’s algorithm exactly

Example: all subsets of \{1,\ldots,n/2\} + all subsets of \{n/2+1,\ldots,n\}
Matching Huffman’s algorithm exactly

Example: all subsets of \{1,...,n/2\} + all subsets of \{n/2+1,...,n\}

Size: $1.4142^n$, best known explicit construction
Matching Huffman’s algorithm exactly

Example: all subsets of \{1,\ldots,n/2\} + all subsets of \{n/2+1,\ldots,n\}

Size: \(1.4142^n\), best known explicit construction

Random construction gives \(1.25^n\), which is optimal!
Matching Huffman’s algorithm exactly

Example: all subsets of \{1,\ldots,n/2\} + all subsets of \{n/2+1,\ldots,n\}

Size: \(1.4142^n\), best known explicit construction

Random construction gives \(1.25^n\), which is optimal!

Construction: choose \(1.25^n\) random sets of every size
Matching Huffman’s algorithm exactly

Example: all subsets of \{1,...,n/2\} + all subsets of \{n/2+1,...,n\}

Size: \(1.4142^n\), best known explicit construction

Random construction gives \(1.25^n\), which is optimal!

Construction: choose \(1.25^n\) random sets of every size

Optimal number of questions for Huffman + \(\varepsilon\): \(n^{O(1/\varepsilon)}\)
What about smooth distributions?
What about smooth distributions?

Huffman worst case $H(\mu)+1$ only obtained when $\mu$ almost constant
What about smooth distributions?

Huffman worst case $H(\mu) + 1$ only obtained when $\mu$ almost constant

What happens when all probabilities in $\mu$ are small?
What about smooth distributions?

Huffman worst case $H(\mu)+1$ only obtained when $\mu$ almost constant.

What happens when all probabilities in $\mu$ are small?

Gallager: cannot go below $H(\mu)+0.086$
What about smooth distributions?

Huffman worst case $H(\mu)+1$ only obtained when $\mu$ almost constant

What happens when all probabilities in $\mu$ are small?

Gallager: cannot go below $H(\mu)+0.086$

Achieved for uniform distributions!
What about smooth distributions?

Huffman worst case $H(\mu) + 1$ only obtained when $\mu$ almost constant.

What happens when all probabilities in $\mu$ are small?

Gallager: cannot go below $H(\mu) + 0.086$.

Achieved for uniform distributions!

What do we get with “<” questions? With “<” and “=” questions?
What about smooth distributions?

Huffman worst case $H(\mu)+1$ only obtained when $\mu$ almost constant

What happens when all probabilities in $\mu$ are small?

Gallager: cannot go below $H(\mu)+0.086$

Achieved for uniform distributions!

What do we get with “<” questions? With “<” and “=” questions?

“<” questions: $H(\mu)+1.086$ [Nakatsu]
What about smooth distributions?

Huffman worst case $H(\mu)+1$ only obtained when $\mu$ almost constant

What happens when all probabilities in $\mu$ are small?

Gallager: cannot go below $H(\mu)+0.086$

Achieved for uniform distributions!

What do we get with “<” questions? With “<” and “=” questions?

“<” questions: $H(\mu)+1.086$ [Nakatsu]

“<” and “=” questions: between $H(\mu)+0.501$ and $H(\mu)+0.586$
Many open questions
Many open questions

• Questions with \( d > 2 \) answers? Mehalel: 1.25 → 1 + \( \frac{(d-1)}{d^{d/(d-1)}} \)
Many open questions

• Questions with $d > 2$ answers? Mehalel: $1.25 \rightarrow 1 + \frac{(d-1)}{d^{\frac{d}{d-1}}}$

• Other models of errors? At most $p$ fraction, at most $q$ fraction in every prefix
Many open questions

- Questions with $d > 2$ answers? Mehalel: $1.25 \rightarrow 1 + \frac{(d-1)}{d^{d/(d-1)}}$

- Other models of errors? At most $p$ fraction, at most $q$ fraction in every prefix

- What happens if we limit worst-case number of questions?
Many open questions

• Questions with \(d>2\) answers? Mehalel: 1.25 \(\rightarrow\) 1 + \((d-1)/d^{d/(d-1)}\)

• Other models of errors? At most \(p\) fraction, at most \(q\) fraction in every prefix

• What happens if we limit worst-case number of questions?

• Fast algorithms for finding optimal binary split trees?
Many open questions

• Questions with $d>2$ answers? Mehalel: $1.25 \rightarrow 1 + (d-1)/d^{d/(d-1)}$

• Other models of errors? At most $p$ fraction, at most $q$ fraction in every prefix

• What happens if we limit worst-case number of questions?

• Fast algorithms for finding optimal binary split trees?

Thank You!