Separations in Query Complexity using Cheat Sheets

Scott Aaronson, Shalev Ben-David, Robin Kothari
Query Complexity

• Fix a Boolean function $f : \{0,1\}^n \rightarrow \{0,1\}$
• How many queries to an unknown binary string $x$ do we need in order to compute $f(x)$?
  • $D(f) = \text{deterministic queries}$
  • $R(f) = \text{randomized queries (with bounded error)}$
  • $Q(f) = \text{quantum queries (with bounded error)}$
Gap Between D and R

• $f(x) = 1$ if $x$ has $2/3$ or more 1s
• $f(x) = 0$ if $x$ has $1/3$ or less 1s
• We assume that $x$ satisfies one of the above conditions

• $R(f)=1$, $D(f) \approx n$
Gap Between R and Q

• Simon’s problem [Simon ‘94]:
  • Suppose x consists of $2^m$ blocks of m bits
  • The position of a block is an m-bit index
  • Assume there’s a hidden m-bit string s such that two blocks are equal iff the xor of their positions is s
  • Goal: find the first bit of s

• Quantum query complexity: $O(\log^2 n)$
• Randomized query complexity: $\Omega\sim(\sqrt{n})$
Gap Between R and Q

• Forrelation [Aaronson, Ambainis 2014]
  • Split x into two parts, each containing $2^m$ blocks of m bits
  • Interpret this as two functions from ${0,1}^m$ to ${0,1}^m$
  • Assume that the first function is either highly correlated with the Fourier transform of the second, or else has near-zero correlation
  • Goal: determine which is the case

• Quantum Query Complexity: 1
• Randomized Query Complexity: $\Omega(\sqrt{n} / \log n)$
Gap between R and Q

- $k$-fold Forrelation:
  - Quantum Query Complexity $O(k)$
  - Conjectured to have randomized query complexity $\Omega(n^{1-1/k})$
- If so, this is the optimal separation
- $k = \log n$ gives $O(\log n)$ vs. $\Omega(n)$
What about total functions?

• [BBCMdW ’98]:

\[ D(f) = O(R(f)^3) \]
\[ D(f) = O(Q(f)^6) \]

• So none of these constructions can work for total functions!
What about total functions?

• Saks, Wigderson ‘86:
  • And-Or tree of depth $\log n$
  • Deterministic query complexity $\Omega(n)$
  • Randomized query complexity $\Theta(n^{0.753})$

• Grover ‘96:
  • OR
  • Randomized query complexity $\Omega(n)$
  • Quantum query complexity $\Theta(\sqrt{n})$
Separations in 2015

- April 4 (Göös, Pitassi, Watson):
  - Introduced the idea of pointer functions
  - Quadratic separation between $D(f)$ and $\deg(f)$
- June 16 (Ambainis, Balodis, Belovs, Lee, Santha, Smotrovs):
  - Quadratic separation between $D(f)$ and $R(f)$
  - Power 4 separation between $D(f)$ and $Q(f)$
  - Many other separations, involving $R_0(f)$ and $Q_E(f)$
- June 26 (B.):
  - Power 2.5 separation between $R(f)$ and $Q(f)$
  - Introduced cheat sheets
- Nov 5 (Aaronson, B., Kothari):
  - Used cheat sheets to reprove many of the other separations
  - Power 4-$o(1)$ separation between $Q(f)$ and approximate degree
Cheat Sheets
Turning partial functions total

• Given a partial function \( f \) that has a good separation, how can we turn it total?
• For concreteness, set \( f \) to be \( f(x) = 1 \) if \( x \) is 2/3 ones, 0 if \( x \) is 2/3 0s (“two-thirds”)
Turning partial functions total

- **Attempt:**

\[
f'(x) = \begin{cases} 
  f(x) & \text{if } x \in \text{Dom}(f) \\
  0 & \text{otherwise}
\end{cases}
\]
Turning partial functions total

• The problem is that the *promise* is difficult for a randomized algorithm to calculate

\[ p_f(x) = \begin{cases} 
1 & \text{if } x \in \text{Dom}(f) \\
0 & \text{otherwise} 
\end{cases} \]
Cheat sheet step 1: Make things easy to certify

- Change the function so that it is easy to certify if an input satisfies the promise
- Also, make it easy to certify whether $f(x)$ is 0 or 1
- But make sure not to decrease $D(f)$!

- **Idea**: Compose with AND-OR

```
  0 0 1 0 1 0 1 0
  1 1 1 1 1 1 0 0
  1 0 1 0 0 0 1 0
  1 0 1 1 1 1 1 1
  0 0 1 0 1 1 1 0
  1 1 1 1 1 1 0 1
  0 1 1 0 0 1 0 1
  1 0 1 1 0 1 1 1
```
Cheat sheet step 1: Make things easy to certify

• Change the function so that it is easy to certify if an input satisfies the promise
• Also, make it easy to certify whether f(x) is 0 or 1
• But make sure not to decrease D(f)!

• Idea: Compose with AND-OR
Cheat sheet step 1: Make things easy to certify

- Change the function so that it is easy to certify if an input satisfies the promise
- Also, make it easy to certify whether f(x) is 0 or 1
- But make sure not to decrease D(f)!

- **Idea**: Compose with AND-OR

```
0 0 1 0 1 0 1 0 1 0
1 1 1 1 1 0 0 0 1 0
1 0 0 0 0 1 0
1 0 1 1 1 1 1 1
0 0 1 0 1 1 1 0
0 0 1 0 1 1 1 0
1 1 1 1 1 1 0 1
0 1 1 0 0 1 0 1
1 0 1 1 0 1 1 1
```
Cheat sheet step 1: Make things easy to certify

- Change the function so that it is easy to certify if an input satisfies the promise
- Also, make it easy to certify whether f(x) is 0 or 1
- But make sure not to decrease D(f)!

- **Idea**: Compose with AND-OR
Properties of the composition

- $D(\text{AND-OR}) = m^2$, $R(\text{AND-OR}) = \Omega(m^2)$
- $D(g) = nm^2$, $R(g) = O(m^2)$
- An input $x$ can be certified to be in the promise of $g$ by certifying all the AND-OR copies (using $nm$ bits)
- This also certifies whether $g(x)$ is 0 or 1
- **But $g$ is still not a total function!**

\[
\begin{array}{cccccccc}
0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\
1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\
\end{array}
\]
Step 2: hide a cheat sheet

• So far:
  • $D(g)$ large
  • $R(g)$ small
  • For every input $x$ in the promise of $g$, there is a small cheat sheet that tells us everything about it

• Next step:
  • change $g$ so it contains the cheat sheet inside it
  • Make sure only a randomized algorithm will be able to find the cheat sheet
Step 2: hide a cheat sheet

Cheat sheet

Input to g

Giant array
Step 3: find the cheat sheet

- We want to let a randomized algorithm find the cheat sheet
- We want a deterministic algorithm to NOT find it
- What can a randomized algorithm do that a deterministic one can’t?
- Solve $g$!

- **Idea:** let $g(x)$ describe where the cheat sheet is
- **Problem:** $g(x)$ is only one bit
- **Solution:** use $100 \log n$ copies of $g$ to describe a position in an array of size $n^{100}$
Step 3: Find the cheat sheet

Array of size $n^{100}$

100 log n Inputs to g
The final function

• Let \( g_{CS} \) be defined by
  • \( g_{CS}(x) = 1 \) if \( x \) has a valid cheat sheet in the right spot of the array
  • \( g_{CS}(x) = 0 \) otherwise
• Then \( g_{CS} \) is a total function!

• \( R(g_{CS}) = ? \)
• Need to compute \( g \) 100 log \( n \) times
  • Each takes \( O(m^2) \)
• Need to check that the cheat sheet is valid
  • There are \( nm \) pointers per copy of \( g \), times 100 log \( n \) copies
  • Each takes \( O(\log nm) \) queries to read, so \( \tilde{O}(nm) \)
• Total is \( nm + m^2 \) (times log factors)
The final function

- Let $g_{CS}$ be defined by
  - $g_{CS}(x) = 1$ if $x$ has a valid cheat sheet in the right spot of the array
  - $g_{CS}(x) = 0$ otherwise
- Then $g_{CS}$ is a total function!

- $D(g_{CS}) = \Omega(nm^2)$
- Blindly searching the array is hopeless
- Must compute $g$ at least once
- Setting $m=n$ gives $D(g_{CS}) \approx n^3$, $R(g_{CS}) \approx n^2$
The final function

Array of size $n^{100}$

100 log $n$ Inputs to $g$
A Super-Grover Speedup
Cheat sheet framework

Array of size $n^{100}$

100 log $n$ Inputs to $g$

$g$

AND-OR

$\ldots$

$\ldots$

$\ldots$

$f$

$\ldots$
Cheat sheet framework

Array of size $n^{100}$

100 log $n$ Inputs to g

$g = \text{Forrelation}$

AND-OR
How many quantum queries?

Verifying certificate for $g$: $n$ queries to read input to $f$, plus $\sqrt{m}\sqrt{n}$ to Grover search over $n$ certificates of size $m$ looking for an error

Total: $\tilde{O}(n + m + \sqrt{m}\sqrt{n}) = \tilde{O}(n)$ if $m = n$
How many random queries?

Array of size $n^{100}$

$g$  $\Omega(m^2\sqrt{n})$

$\Omega(m^2\sqrt{n}/\log n)$

$100 \log n$ Inputs to $g$

$f = \text{Forrelation}$

AND-OR

$\Omega(n/\log n)$

$\Omega(m^2)$

Array of size $n^{100}$

Total: $\Omega(n^{2.5})$ if $m = n$
Conclusion: power 2.5 speedup

Array of size $n^{100}$

$g$

$100 \log n$ Inputs to $g$

$f = \text{Forrelation}$

$\text{AND-OR}$
Summary

• Power 2.5 separation between randomized and quantum query complexity
• Becomes power 3 separation if we can show a log n vs. n separation in the promise setting
• Best known upper bound is 6
Communication Complexity?

- We want to lift this to communication complexity
- We could use a measure that
  - Lower-bounds $R(f)$
  - Give a good lower bound for forrelation (or Simon’s)
  - Composes (with AND-OR)
  - Is preserved under addition of cheat sheets
  - Lifts to communication lower bound

- Alternatively, lift to communication complexity before adding cheat sheets
- Prove a lower bound on $R$ for cheat sheet functions in communication complexity
# More Complexity Measures

<table>
<thead>
<tr>
<th></th>
<th>$D$</th>
<th>$R_0$</th>
<th>$R$</th>
<th>$C$</th>
<th>$RC$</th>
<th>$bs$</th>
<th>$Q_E$</th>
<th>$\text{deg}$</th>
<th>$Q$</th>
<th>$\tilde{\text{deg}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D$</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>4, 6</td>
<td>4, 6</td>
<td></td>
</tr>
<tr>
<td>$R_0$</td>
<td>1, 1</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>4, 6</td>
<td>4, 6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$R$</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td></td>
</tr>
<tr>
<td>$C$</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td></td>
</tr>
<tr>
<td>$RC$</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td></td>
</tr>
<tr>
<td>$bs$</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td></td>
</tr>
<tr>
<td>$Q_E$</td>
<td>1, 1</td>
<td>1.3267, 2</td>
<td>1.3267, 3</td>
<td>1.3267, 3</td>
<td>1.3267, 3</td>
<td>1.3267, 3</td>
<td>1.3267, 3</td>
<td>1.3267, 3</td>
<td>1.3267, 3</td>
<td></td>
</tr>
<tr>
<td>$\text{deg}$</td>
<td>1, 1</td>
<td>1.3267, 2</td>
<td>1.3267, 3</td>
<td>1.3267, 3</td>
<td>1.3267, 3</td>
<td>1.3267, 3</td>
<td>1.3267, 3</td>
<td>1.3267, 3</td>
<td>1.3267, 3</td>
<td></td>
</tr>
<tr>
<td>$Q$</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td></td>
</tr>
<tr>
<td>$\tilde{\text{deg}}$</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td></td>
</tr>
</tbody>
</table>

- **New separations**
- **Separations we reprove**
# More Complexity Measures

<table>
<thead>
<tr>
<th></th>
<th>$D$</th>
<th>$R_0$</th>
<th>$R$</th>
<th>$C$</th>
<th>RC</th>
<th>bs</th>
<th>$Q_E$</th>
<th>$\text{deg}$</th>
<th>$Q$</th>
<th>$\tilde{\text{deg}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D$</td>
<td>2, 2</td>
<td>2, 3</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 3</td>
<td>2, 3</td>
<td>2, 3</td>
<td>4*, 6</td>
<td>4*, 6</td>
<td></td>
</tr>
<tr>
<td>$R_0$</td>
<td>1, 1</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 3</td>
<td>2, 3</td>
<td>2, 3</td>
<td>4*, 6</td>
<td>4*, 6</td>
<td></td>
</tr>
<tr>
<td>$R$</td>
<td>1, 1</td>
<td>1, 1</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 3</td>
<td>1.5, 3</td>
<td>2, 3</td>
<td>2.5, 6</td>
<td>4*, 6</td>
<td></td>
</tr>
<tr>
<td>$C$</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>1.1527, 3</td>
<td>$\log_3 6, 3$</td>
<td>2, 4</td>
<td>2, 4</td>
<td></td>
</tr>
<tr>
<td>RC</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1.5, 2</td>
<td>1.1527, 3</td>
<td>$\log_3 6, 3$</td>
<td>2, 2</td>
<td>2, 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bs</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1.1527, 3</td>
<td>$\log_3 6, 3$</td>
<td>2, 2</td>
<td>2, 2</td>
<td></td>
</tr>
<tr>
<td>$Q_E$</td>
<td>1, 1</td>
<td>1.3267, 2</td>
<td>1.3267, 3</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 3</td>
<td>2, 3</td>
<td>4*, 6</td>
<td>4*, 6</td>
<td></td>
</tr>
<tr>
<td>$\text{deg}$</td>
<td>1, 1</td>
<td>1.3267, 2</td>
<td>1.3267, 3</td>
<td>2, 2</td>
<td>2, 2</td>
<td>1, 1</td>
<td>2, 6</td>
<td>2, 6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Q$</td>
<td>1, 1</td>
<td>1, 1</td>
<td>2, 2</td>
<td>2, 2</td>
<td>$\text{Th. 3}$</td>
<td>2, 3</td>
<td>2, 3</td>
<td>$4^*, 6$</td>
<td>$\text{Th. 2}$</td>
<td></td>
</tr>
<tr>
<td>$\tilde{\text{deg}}$</td>
<td>1, 1</td>
<td>1, 1</td>
<td>7/6, 2</td>
<td>7/6, 3</td>
<td>7/6, 3</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td></td>
</tr>
</tbody>
</table>

- **New separations**
- **Separations we reprove**
Approximate Degree

- Lower bound for $Q$
- Previous separation: $1.3$ (Ambainis 2003)
- This work: $4 - o(1)$
- Most complicated function used in query complexity
  - At the time, at least...
Unambiguous Certificates

A set of unambiguous 1-certificates is a set of 1-certificates for f such that
- Any two of them contradict each other
- Any 1-input to f contains one of them

Example: f = OR₄

```
1___  01___ 001_  0001
```

Let UC⁽¹⁾(f) be the size of the largest certificate in the best choice of unambiguous 1-certificates
Polynomials from $\text{UC}^{(1)}$

- Let $S$ be a set of unambiguous 1-certificates for $f$
- For any certificate $c$ in $S$, there is a low-degree polynomial $p_c$ for checking if the input contains the certificate
  - $p_c(x) = 1$ iff $x$ contains $c$
  - $\deg(p_c) = |c|$
- Add up $p_c$ for all $c$ in $S$ to get a polynomial $p$
- Each 1-input contains exactly one certificate in $S$
  - $p(x) = 1$ if $f(x) = 1$
  - $p(x) = 0$ if $f(x) = 0$
- Conclusion: $\deg(f) \leq \text{UC}^{(1)}(f)$
Approximate degree from UC\(^{(1)}\)

- Let S be a set of unambiguous 1-certificates for f.
- Suppose that for any certificate c in S, there is a low-degree polynomial \(p_c\) for checking if the input contains the certificate:
  - \(p_c(x) \geq 2/3\) if x contains c
  - \(p_c(x) = 0\) if x does not contain c
- Add up \(p_c\) for all c in S to get a polynomial \(p\).
- Each 1-input contains exactly one certificate in S:
  - \(p(x) \geq 2/3\) if \(f(x)=1\)
  - \(p(x) = 0\) if \(f(x)=0\)
- Conclusion: \(\text{adeg}(f) \leq \text{Quantum complexity of checking UC}^{(1)}\) certificates.
Cheat Sheets and UC(1)

- **Observation**: $\text{UC}^{(1)}(f_{\text{CS}}) \approx C(f)$
- The unambiguous 1-certificates will be the correct cheat sheet cell and all the certificates it points to

- **Implication**: $\deg(f_{\text{CS}}) \leq C(f)$
- So adding a cheat sheet to AND-OR gives a quadratic gap between $\deg$ and $R$
- Since certificates for AND-OR can be checked in $\sqrt{m}$ quantum queries, this gives a power 4 separation between $\text{adeg}$ and $R$

- What about $Q$?
k-Sum

- Are there $k$ elements summing to 0 mod $M$?
- Set $k = \log n$
- $Q \approx n$, $C^{(1)} \approx \text{polylog } n$ (Belovs and Špalek 2013)
Block k-Sum

- Split input into blocks; a block is balanced if it has the same number of 0s and 1s
- Balanced blocks represent numbers
- If there are \( \log n \) balanced blocks summing to 0 mod M and all other blocks have at least as many 1s as 0s, \( f(x) = 1 \)
- \( Q \) is large, all certificates use almost only 1s
Q \approx C^2
RecBKK
RecBKK_{CS}

\[ Q \approx \text{approxdeg}^{4-o(1)} \]
## Open Problems

<table>
<thead>
<tr>
<th></th>
<th>$D$</th>
<th>$R_0$</th>
<th>$R$</th>
<th>$C$</th>
<th>$RC$</th>
<th>$bs$</th>
<th>$Q_E$</th>
<th>$\text{deg}$</th>
<th>$Q$</th>
<th>$\tilde{\text{deg}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D$</td>
<td></td>
<td>[ABB+15]</td>
<td>[ABB+15]</td>
<td>[ABB+15]</td>
<td>$\Lambda \circ \vee$</td>
<td>$\Lambda \circ \vee$</td>
<td>$\Lambda \circ \vee$</td>
<td>2, 3</td>
<td>[ABB+15]</td>
<td>[ABB+15]</td>
</tr>
<tr>
<td>$R_0$</td>
<td>1, 1</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>$\Lambda \circ \vee$</td>
<td>$\Lambda \circ \vee$</td>
<td>$\Lambda \circ \vee$</td>
<td>2, 3</td>
<td>[GPW15]</td>
<td>[ABB+15]</td>
</tr>
<tr>
<td>$R$</td>
<td>1, 1</td>
<td>2, 2</td>
<td>$\Lambda \circ \vee$</td>
<td>$\Lambda \circ \vee$</td>
<td>$\Lambda \circ \vee$</td>
<td>$\Lambda \circ \vee$</td>
<td>2, 3</td>
<td>[ABB+15]</td>
<td>3, 6</td>
<td>4*, 6</td>
</tr>
<tr>
<td>$C$</td>
<td>1, 1</td>
<td>1, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>$\Lambda \circ \vee$</td>
<td>$\Lambda \circ \vee$</td>
<td>2.5, 6</td>
<td>Th. 1</td>
<td>[ABB+15]</td>
<td>[ABB+15]</td>
</tr>
<tr>
<td>$RC$</td>
<td>1, 1</td>
<td>1, 1</td>
<td>2, 2</td>
<td>2, 2</td>
<td>$\Lambda \circ \vee$</td>
<td>$\Lambda \circ \vee$</td>
<td>2, 4</td>
<td>2, 4</td>
<td>[NW95]</td>
<td>[NW95]</td>
</tr>
<tr>
<td>$bs$</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>2, 2</td>
<td>$\Lambda \circ \vee$</td>
<td>$\Lambda \circ \vee$</td>
<td>2, 4</td>
<td>2, 4</td>
<td>[NW95]</td>
<td>[NW95]</td>
</tr>
<tr>
<td>$Q_E$</td>
<td>1, 1</td>
<td>1.3267, 2</td>
<td>1.3267, 3</td>
<td>1.3267, 3</td>
<td>$\Lambda \circ \vee$</td>
<td>$\Lambda \circ \vee$</td>
<td>2, 3</td>
<td>4*, 6</td>
<td>Th. 2</td>
<td></td>
</tr>
<tr>
<td>$\text{deg}$</td>
<td>1, 1</td>
<td>1.3267, 2</td>
<td>1.3267, 3</td>
<td>1.3267, 3</td>
<td>$\Lambda \circ \vee$</td>
<td>$\Lambda \circ \vee$</td>
<td>2, 4</td>
<td>Th. 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Q$</td>
<td>1, 1</td>
<td>1, 1</td>
<td>2, 2</td>
<td>2, 2</td>
<td>$\Lambda \circ \vee$</td>
<td>$\Lambda \circ \vee$</td>
<td>2, 4</td>
<td>Th. 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\tilde{\text{deg}}$</td>
<td>1, 1</td>
<td>1, 1</td>
<td>7/6, 2</td>
<td>$\Lambda \circ \vee$</td>
<td>$\Lambda \circ \vee$</td>
<td>$\Lambda \circ \vee$</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
</tr>
</tbody>
</table>

Note: [ABB+15], [GPW15], [ABB+15], [ABB+15], [ABB+15], [NW95], [Amb13], [GSS13], [Amb13], [NW95], [NW95].
## Open Problems

<table>
<thead>
<tr>
<th></th>
<th>(D)</th>
<th>(R_0)</th>
<th>(R)</th>
<th>(C)</th>
<th>(RC)</th>
<th>(bs)</th>
<th>(Q_E)</th>
<th>(\text{deg})</th>
<th>(Q)</th>
<th>(\text{deg})</th>
</tr>
</thead>
<tbody>
<tr>
<td>(D)</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2*</td>
<td>2*</td>
<td>2, 3</td>
<td>2, 3</td>
<td>4*, 6</td>
<td>4*, 6</td>
<td></td>
</tr>
<tr>
<td>(R_0)</td>
<td>1, 1</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2*</td>
<td>2*</td>
<td>2, 3</td>
<td>2, 3</td>
<td>4*, 6</td>
<td>4*, 6</td>
<td></td>
</tr>
<tr>
<td>(R)</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1.5, 3</td>
<td>2, 3</td>
<td>2.5, 6</td>
<td>Th. 1</td>
<td></td>
</tr>
<tr>
<td>(C)</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1.527, 3</td>
<td>log_3 6, 3</td>
<td>2, 4</td>
<td>2, 4</td>
<td></td>
</tr>
<tr>
<td>(RC)</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1.527, 3</td>
<td>log_3 6, 3</td>
<td>2, 2</td>
<td>2, 2</td>
<td></td>
</tr>
<tr>
<td>(bs)</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1.527, 3</td>
<td>log_3 6, 3</td>
<td>2, 4</td>
<td>2, 4</td>
<td></td>
</tr>
<tr>
<td>(Q_E)</td>
<td>1, 1</td>
<td>1.3267, 2</td>
<td>1.3267, 3</td>
<td>2*</td>
<td>2*</td>
<td>2, 3</td>
<td>2, 3</td>
<td>4*, 6</td>
<td>Th. 2</td>
<td></td>
</tr>
<tr>
<td>(\text{deg})</td>
<td>1, 1</td>
<td>1.3267, 2</td>
<td>1.3267, 3</td>
<td>2*</td>
<td>2*</td>
<td>2, 3</td>
<td>2, 6</td>
<td>Th. 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Q)</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>2, 2</td>
<td>2*</td>
<td>2*</td>
<td>2*</td>
<td>4*, 6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\text{deg})</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>7/6, 2</td>
<td>7/6, 3</td>
<td>7/6, 3</td>
<td>1, 1</td>
<td>1, 1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notes:
- [ABB+15] refers to references or citations related to the problems.
- \(\text{deg}\) and \(Q\) are likely related to degree and quantity in the context of these problems.
# Open Problems

<table>
<thead>
<tr>
<th>$D$</th>
<th>$R_0$</th>
<th>$R$</th>
<th>$C$</th>
<th>$RC$</th>
<th>$bs$</th>
<th>$Q_E$</th>
<th>$\text{deg}$</th>
<th>$Q$</th>
<th>$\tilde{\text{deg}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 3</td>
<td>3, 6</td>
<td>4*, 6</td>
</tr>
<tr>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>2, 2</td>
<td>1.527, 3</td>
<td>2, 4</td>
<td>2, 4</td>
</tr>
<tr>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1.527, 3</td>
<td>2, 2</td>
<td>2, 2</td>
</tr>
<tr>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1.527, 3</td>
<td>2, 2</td>
<td>2, 2</td>
</tr>
<tr>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1, 1</td>
<td>1.527, 3</td>
<td>2, 2</td>
<td>2, 2</td>
</tr>
</tbody>
</table>

1. **$D$** represents the domain of the problem.
2. **$R_0$** represents the initial relation.
3. **$R$** represents the refined relation.
4. **$C$** represents the condition.
5. **$RC$** represents the refined condition.
6. **$bs$** represents the base set.
7. **$Q_E$** represents the query.
8. **$\text{deg}$** represents the degree of the problem.
9. **$Q$** represents the query.
10. **$\tilde{\text{deg}}$** represents the adjusted degree of the problem.

Notes:
- [ABB+15]: Reference [ABB+15]
- [GPW15]: Reference [GPW15]
- [GJPW15]: Reference [GJPW15]
Thanks