ECE 50863
Project 3 Report Template

Name and email: Yuan-Yao Lou / yylou@purdue.edu
Collaborator (if applicable):

Submit as a pdf, labeled <firstname.lastname.Project3.first.pdf>,
<firstname.lastname.Project3.final.pdf>

Collaborations should be restricted to brainstorming ideas, and sharing experiences, but each student should implement her/his own code, and the report should be individual

Instruction

- Please follow this template for all milestones, but update the material with each milestone.
- Answers should be typed. Handwritten documents are not permitted.
- Your report should be submitted in pdf format only.
- Please keep your answers to the point.
- **Graphs should be carefully plotted, with the X and Y axis clearly labeled with appropriate legends.**
- **For a slide with a graph have a 1 line “take-home” message (what does the result show?)**
- Avoid 2 graphs on the same slide to ensure clarity.

Base Algorithm 1

- What base algorithm did you implement? **RobustMPC**
- Present a bulleted list of the key details of the implementation of the base algorithm. The description should be self-sufficient to see all your key decisions, but please keep it succinct.
  - **Calculate QoE score of total quality** for all possible quality sequences
  - **Calculate QoE score of quality variation** for all possible quality sequences
  - **Use "previous_throughput" as the throughput** for the 5 look-ahead windows
  - **Get chunk size sequences for all possible quality sequences** by "quality_bitrates" & "upcoming_quality_bitrates"
  - **Calculate download time** by the throughput and chunk size sequences
  - **Calculate rebuffer time** by download time and "buffer_seconds_until_empty"
  - **Calculate the composite QoE score** for all possible quality sequences to make the decision

Variant of Base Algorithm 1

- Discuss the variant of the base algorithm that you implemented. If a standard variant (BBA-0 or BBA-1, FastMPC), this can be brief. But if a more novel variant, expand a bit more on what the variant seeks to achieve, and provide more details.
  - When calculating the QoE score of quality variation for all possible quality sequences, I also **consider the previous chunk's quality**
  - Since rebuffer time has the highest penalty coefficient, **I add a new metric "amount of client buffer" into consideration when choosing the quality of the next chunk**
  - **I extend the look-ahead window size to 9** for calculating the QoE score for all possible quality sequences (not choosing 10 due to much longer execution time)
  - By setting the look-ahead window size to ~10, **I set the QoE coefficient of "amount of client buffer" to 2** (same as the total quality). This is because the MAX client buffer size now approximately equals MAX chunk quality (i.e., [3, 3, 3, 3, 3, 3, 3, 3, 3]).

Results: Base Algorithm 1 and Variant

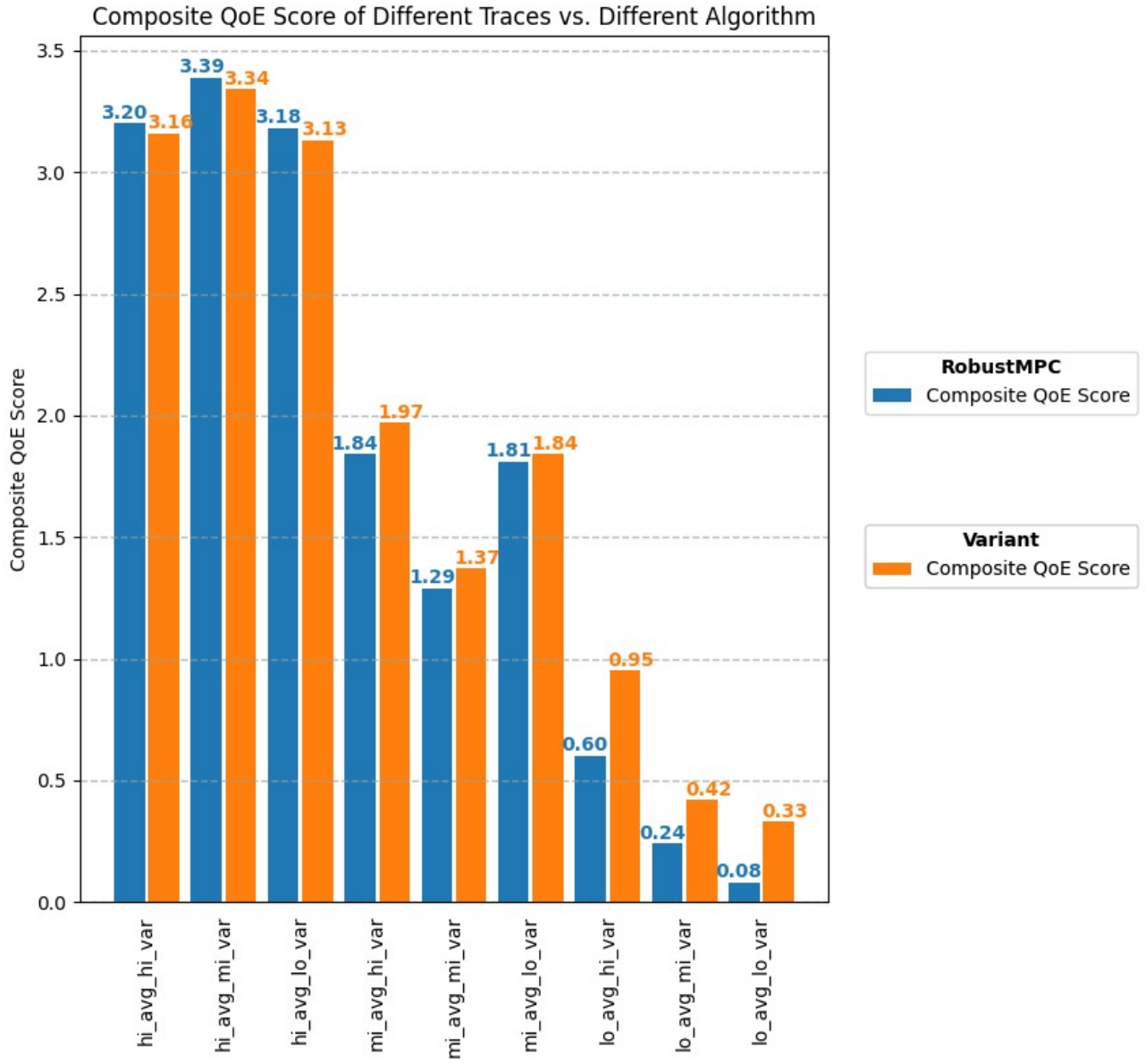
- Present **comparison results of BaseAlgorithm1 and its variant** on the set of traces/configurations that we provided.
- Please give some thought to **what graphs to present** in as compact yet informative/insightful a manner as possible.
- Reporting the composite score is useful, but it may be helpful to also **report other metrics especially when they can help provide more insights.**

Composite Score

The variant performs slightly worse than the base algorithm (RobustMPC) in “hi_avg_*” traces.

The variant performs slightly better than the base algorithm (RobustMPC) in “mi_avg_*” traces.

The variant performs significantly better than the base algorithm (RobustMPC) in “lo_avg_*” traces.

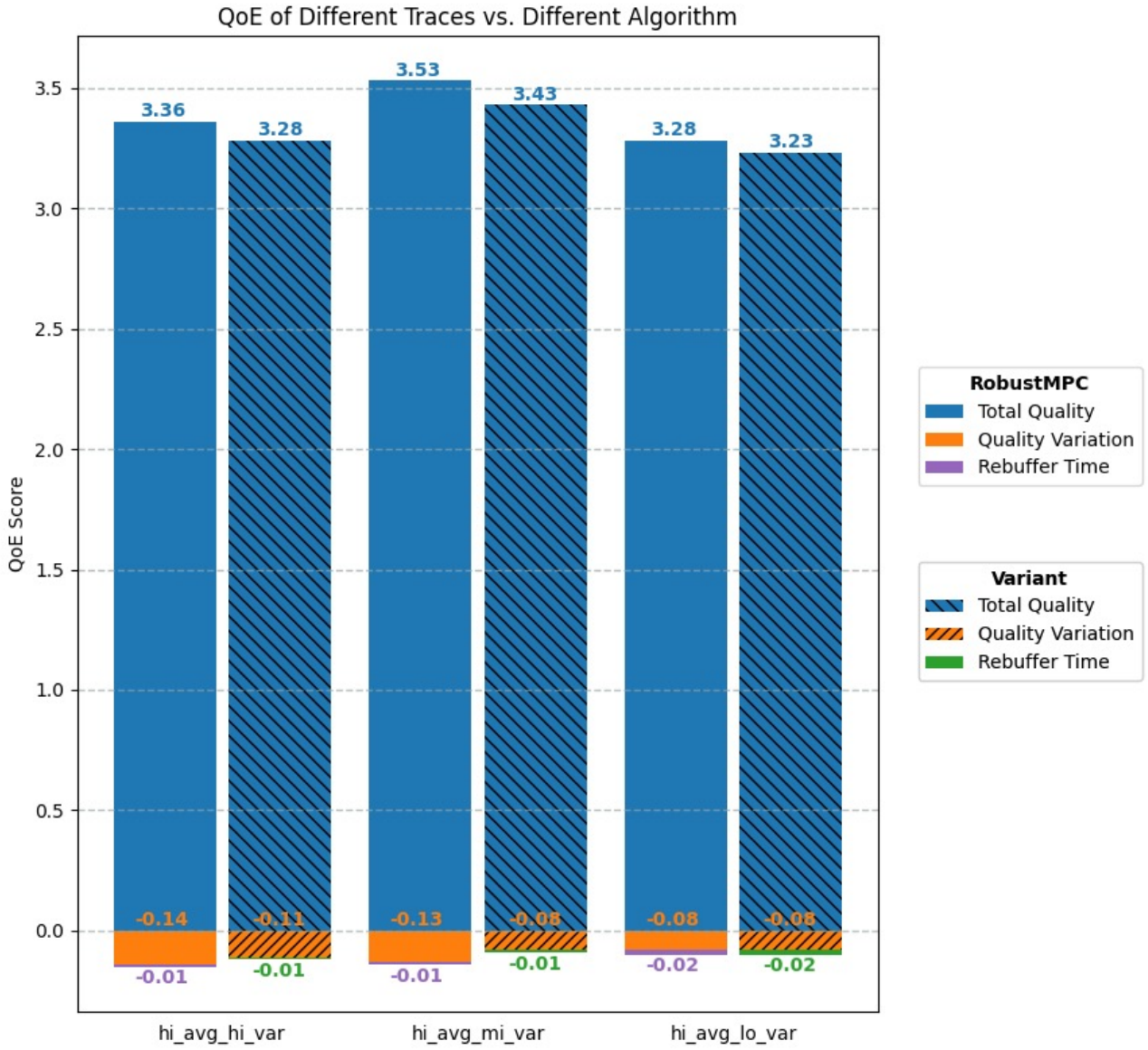


Individual Score: “hi_avg_*” traces

The variant performs slightly worse than the base algorithm (RobustMPC).

The variant gets:

- Lower QoE score of total quality
- Slightly higher QoE score of quality variation

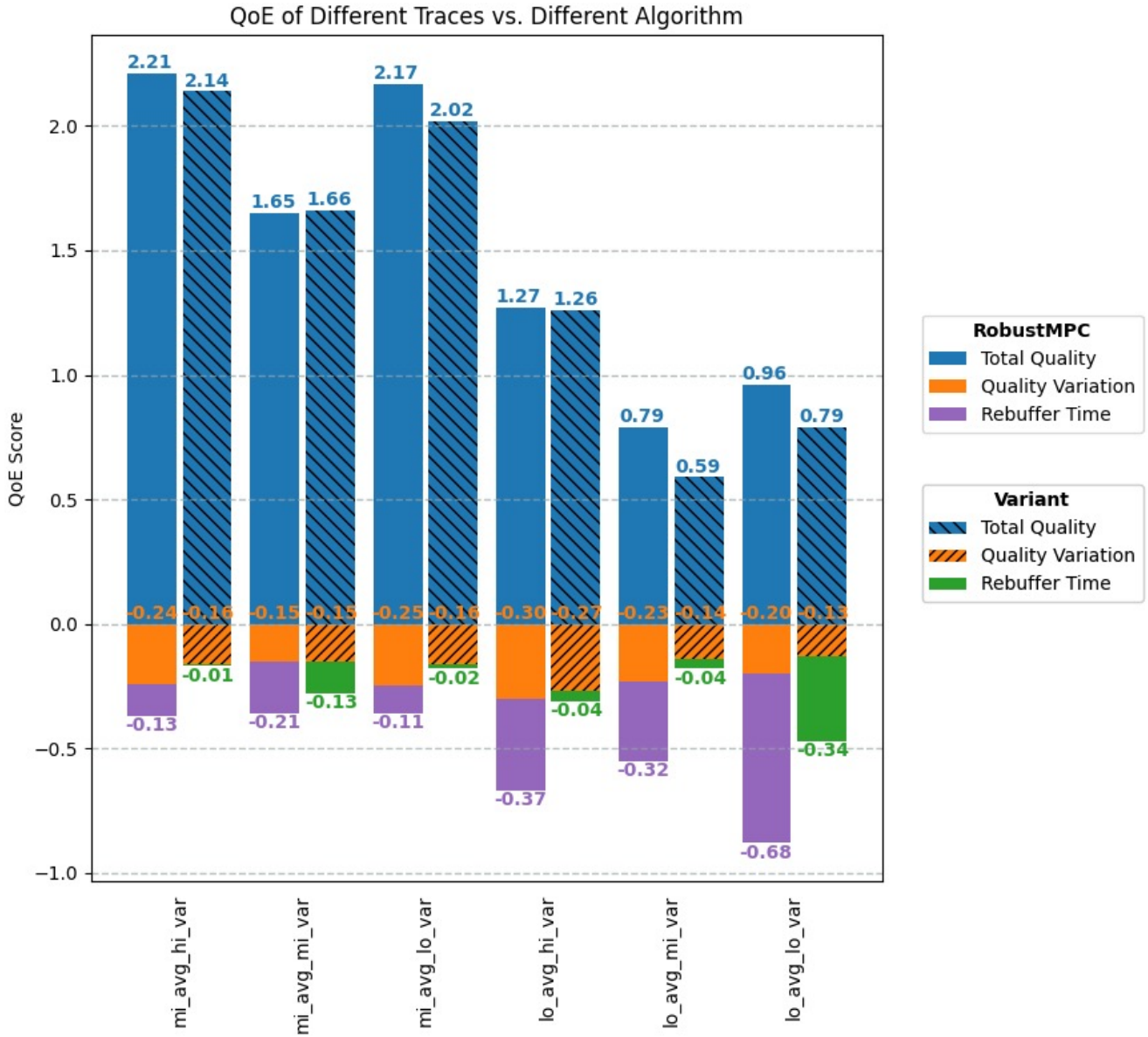


Individual Score: “lo_avg_* & mi_avg_*” traces

The variant performs better than the base algorithm (RobustMPC).

The variant gets:

- Lower QoE score of total quality
- Slightly higher QoE score of quality variation
- Much higher QoE score of rebuffer time



Results Discussion

- What are your conclusions?
  - Does an algorithm (or variant) **generally work better in all cases?**
  - Do the algorithms have **trade-offs,** with one working better in certain environments than others?
  - Do some algorithms **favor one metric (e.g. quality) more than others?**
  - In **what settings does an algorithm perform the best or the worst, and why?**
- Please do not just say “Algorithm 1 performs better In Trace 1 and 2, while Algorithm 2 performs better in Traces 3 to 5”. Instead, seek to generalize and see what characteristics those traces have, and try to connect it to how the algorithm works.

Results Discussion

- **Does an algorithm (or variant) generally work better in all cases? Do the algorithms have trade-offs, with one working better in certain environments than others?**

  The algorithms do have trade-offs regarding different traces.

  While the variant performs slightly worse than the base algorithm in “hi_avg_*” traces, the variant performs better than the base algorithm in “mi_avg_*” traces and “lo_avg_*” traces.

Results Discussion

- **Do some algorithms favor one metric (e.g. quality) more than others?**

  No, the QoE score of almost all metrics decreases when the average throughput decreases. And no matter for the base algorithm or the variant, this trend remains the same.

- **In what settings does an algorithm perform the best or the worst, and why?**

  In high average throughput traces, both the base algorithm and variant perform the best.

  This is because, under higher throughput, both algorithms will have shorter download time which leads to more remaining client buffers and a lower frequency of rebuffering events. Hence, when processing all possible quality sequences, due to the few rebuffer events, both algorithms will prefer selecting the sequence with higher total quality and less quality variation (e.g., [3, 3, 3, 3, 3, …], [2, 3, 3, 3, 3, …], …).

For interim report, you may stop here

For final report, update slide deck to..

- Discuss the details of the second algorithm implemented
- Discuss the variant of the second algorithm you implemented: rationale for it and details.
- Present results first comparing the second algorithm and its variant.
- Then, present results comparing the better performing variant of the first algorithm, and the better performing variant of the second algorithm (you may compare all algorithms and their variants if the results can be presented compactly).
- Update your discussions of results in the interim report to cover both algorithms and variants

Other open-ended explorations

- Discuss any other variants you may have implemented/explored and summarize results.
- If you came up with additional traces of your own to better distinguish different algorithms, discuss the rationale, and what you found.