SEA 2018 report

Contents

1 Experimental setup

1.1 Time measurements

We run a program k times with and without the optimization and recorded the sum construction time of the MS and RUNS vectors. The plots report (median, with quartile ranges) of the speedup of each optimized time $t_i^{\texttt{opt}}$ relative to the average non-optimized time in the construction time of the MS vector. In other words

$$d^{(i)} = \frac{\bar{t}_{\texttt{non\_opt}}}{t^{(i)}_{\texttt{opt}}}$$

with $\bar{t}_{\texttt{non\_opt}} = 1/n \sum t^{(i)}_{\texttt{non\_opt}}$, and $i = 1, \ldots, k$.

The boxplots report the raw times.

2 WL tests

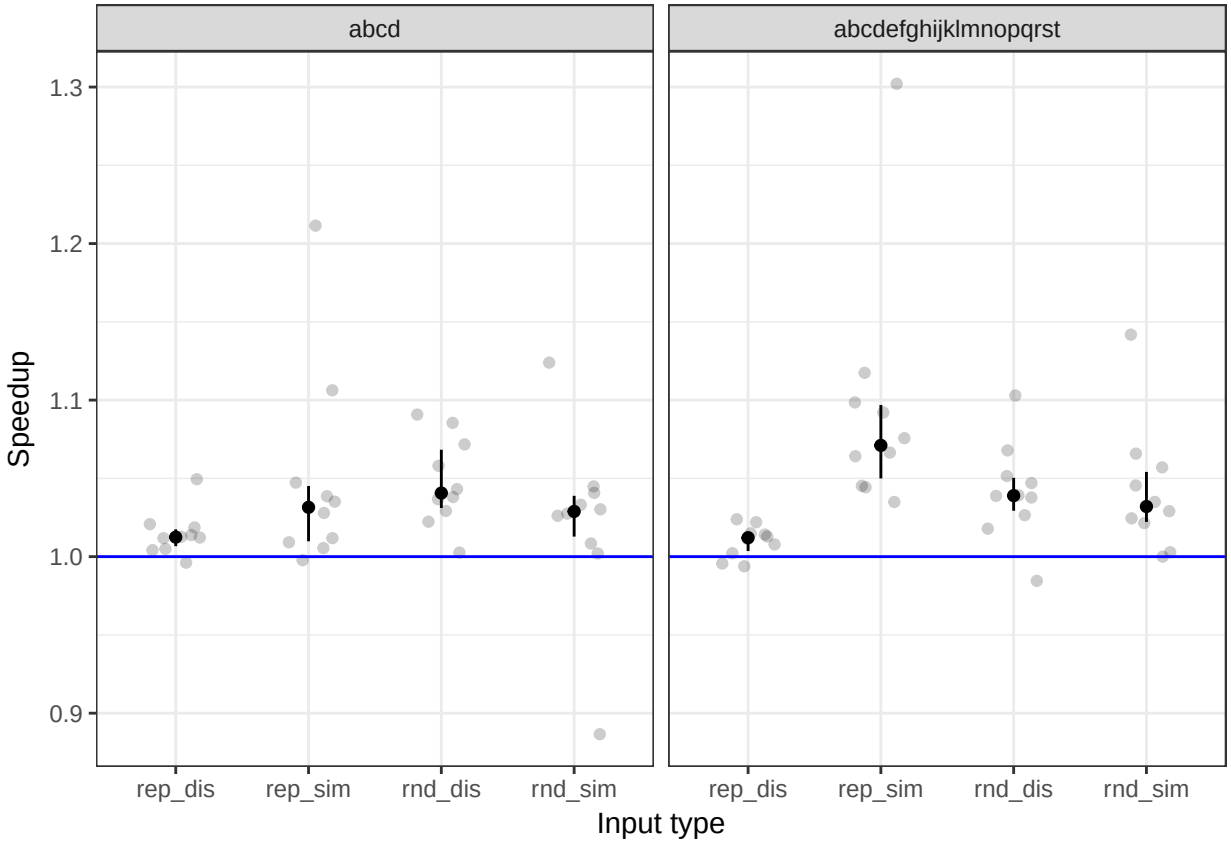
2.1 Input data

We perform tests on the Weiner Link optimizations on 4 types of input.

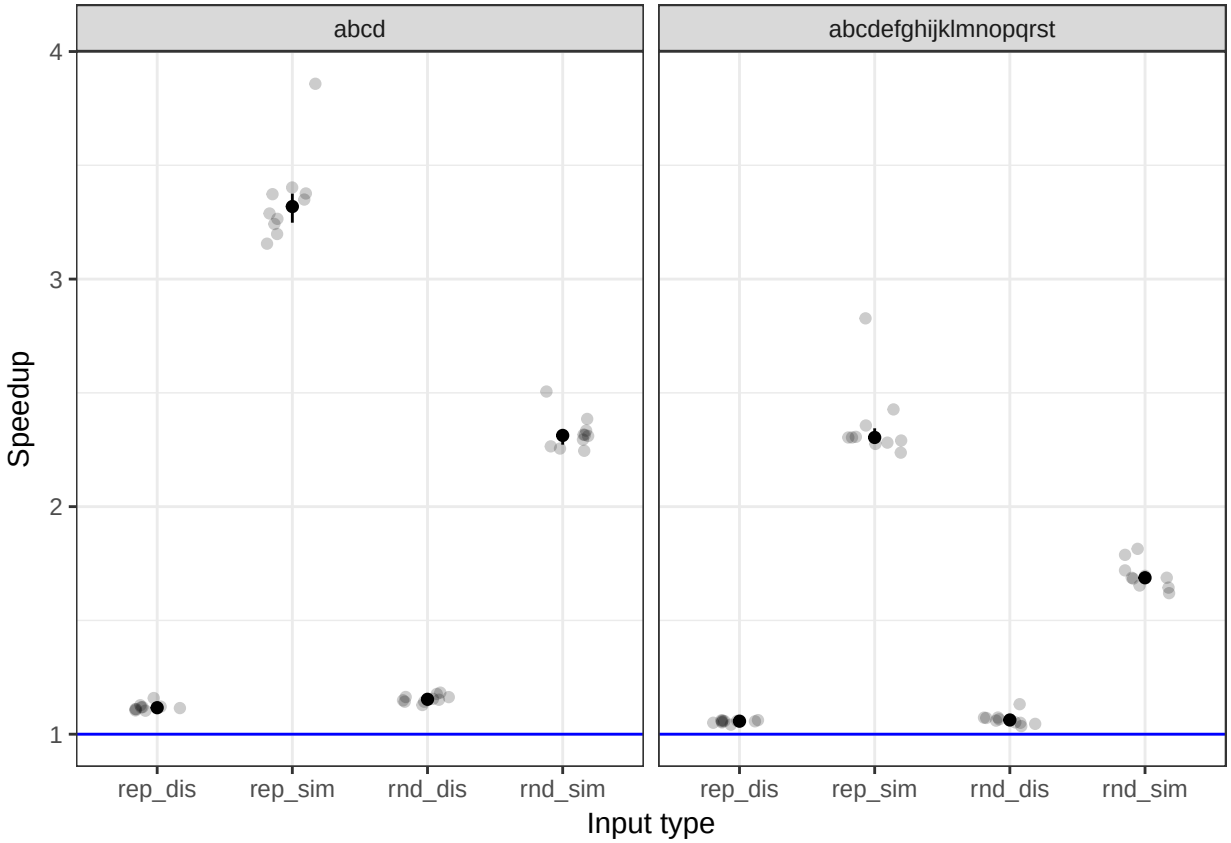
- Index string with repeats, query string random (code: `rep_dis`)
- Index string with repeats, query string similar to index (code: `rep_sim`)
- Index string random, query string random (code: `rnd_dis`)
- Index string random, query string similar to index (code: `rnd_sim`)

Further, we generate all of the above input data for two alphabet sizes: $\Sigma_1| = 4$ and $\Sigma_2| = 20$. For all input types, the index string is of length 100MB and the query 500KB.

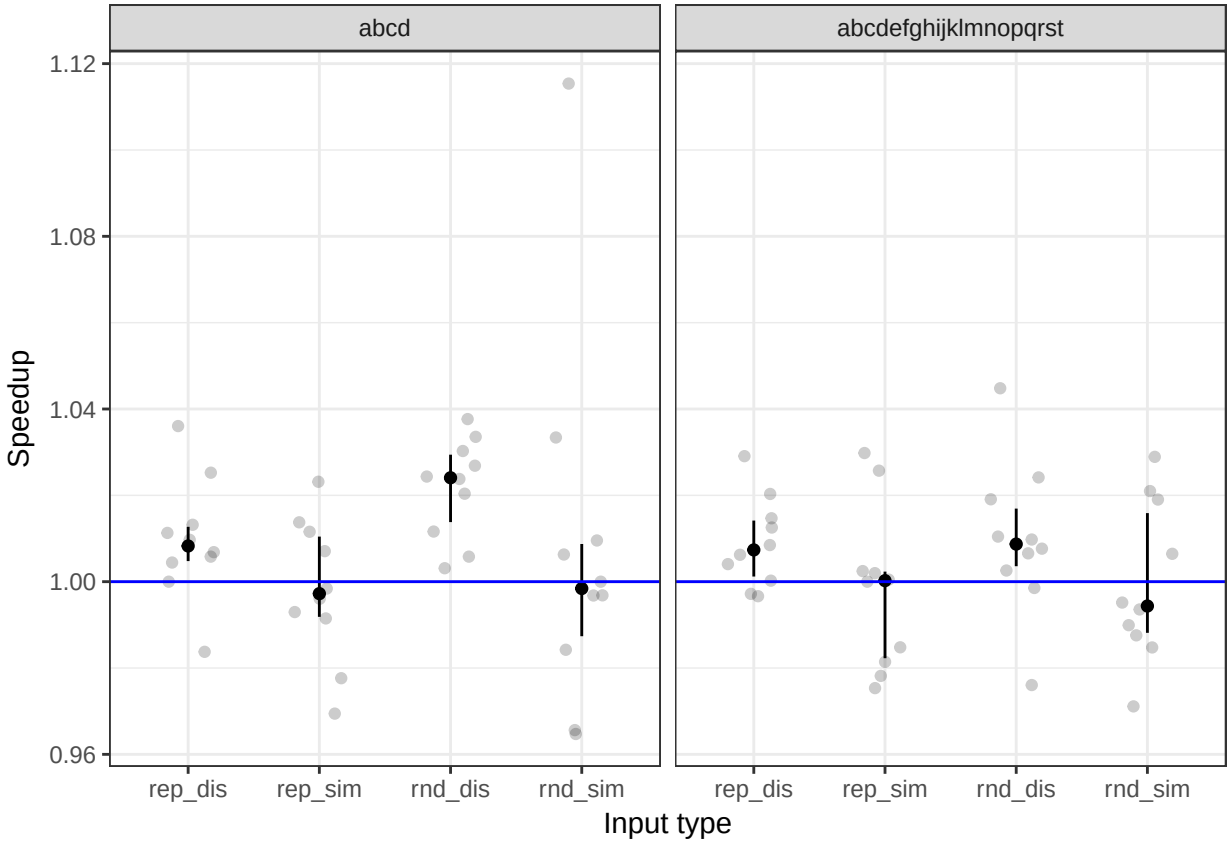
2.2 Double rank versus single rank



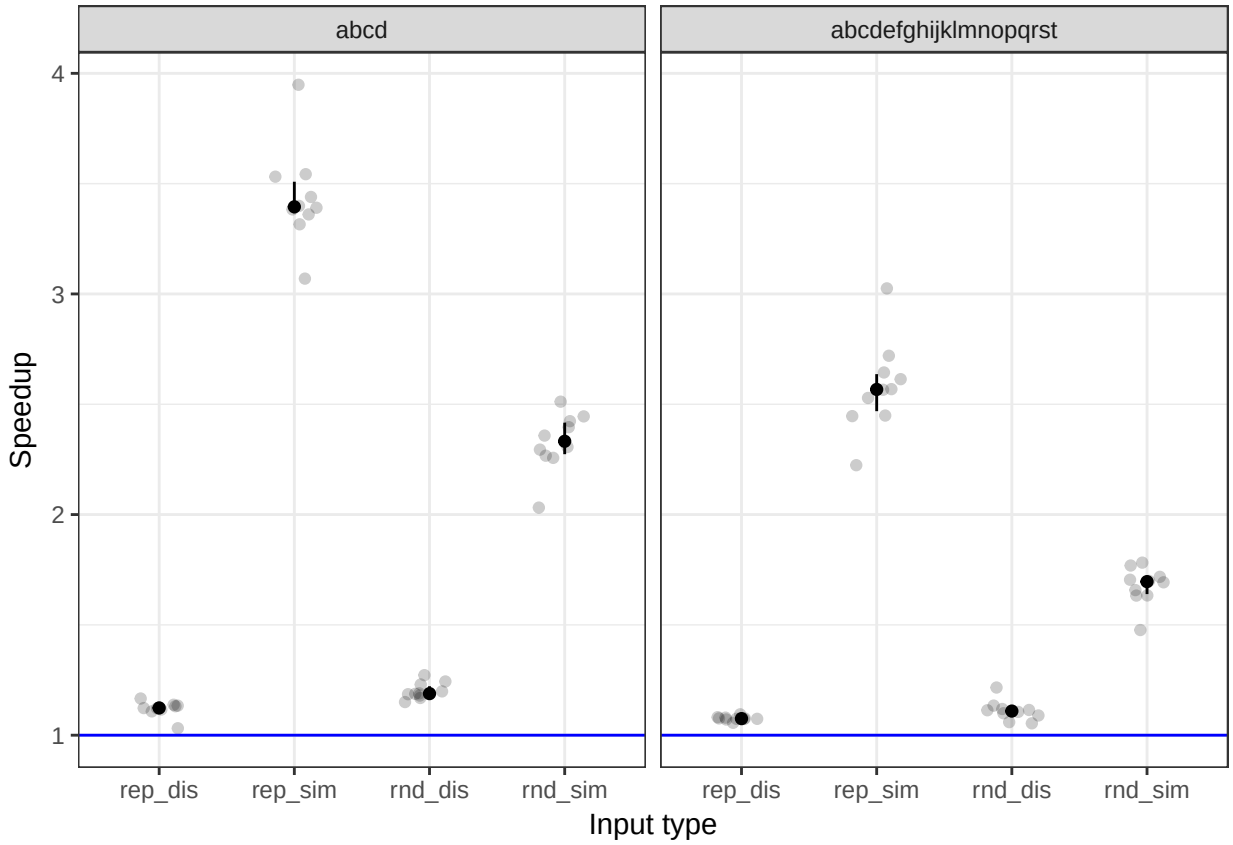
2.3 Lazy versus nonlazy



2.4 Double rank and fail versus double rank

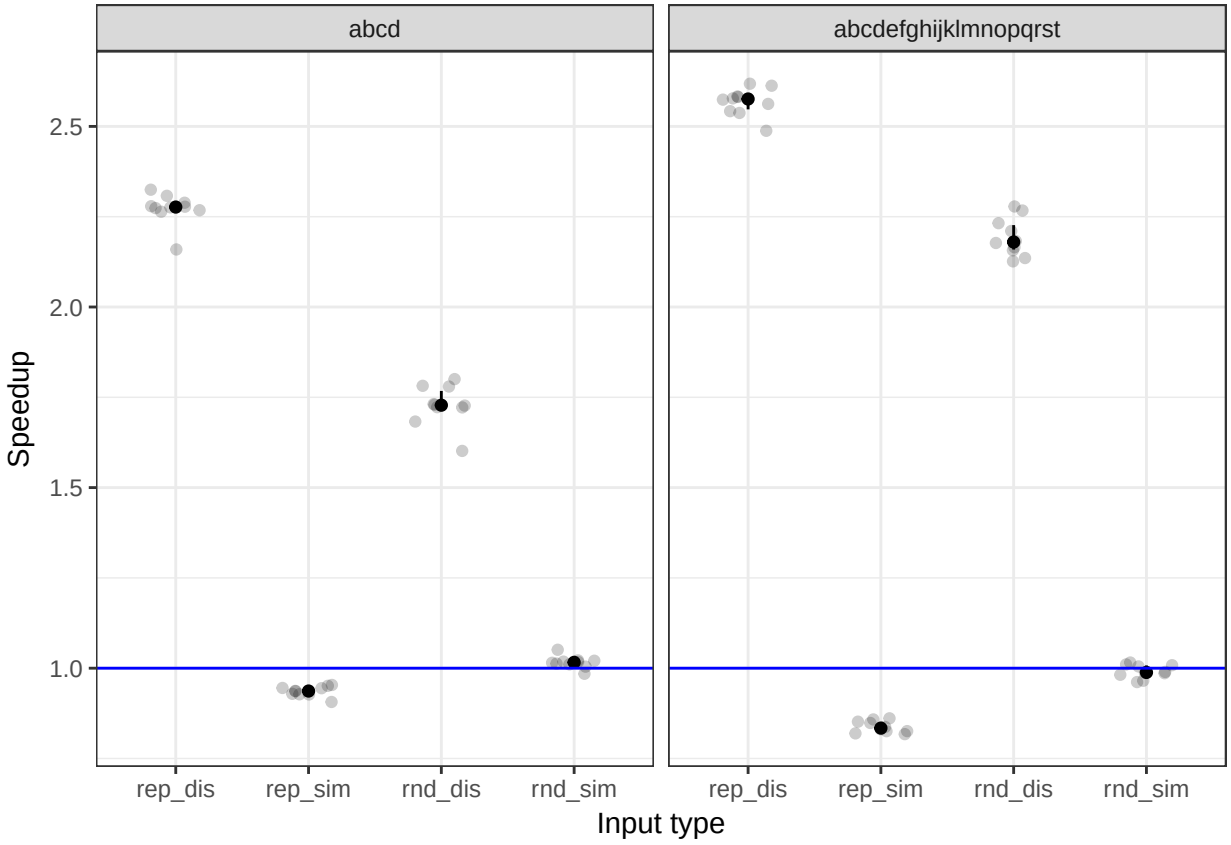


2.5 Double rank, Lazy, and Rank and fail versus Single rank, nonlazy, and nonfail

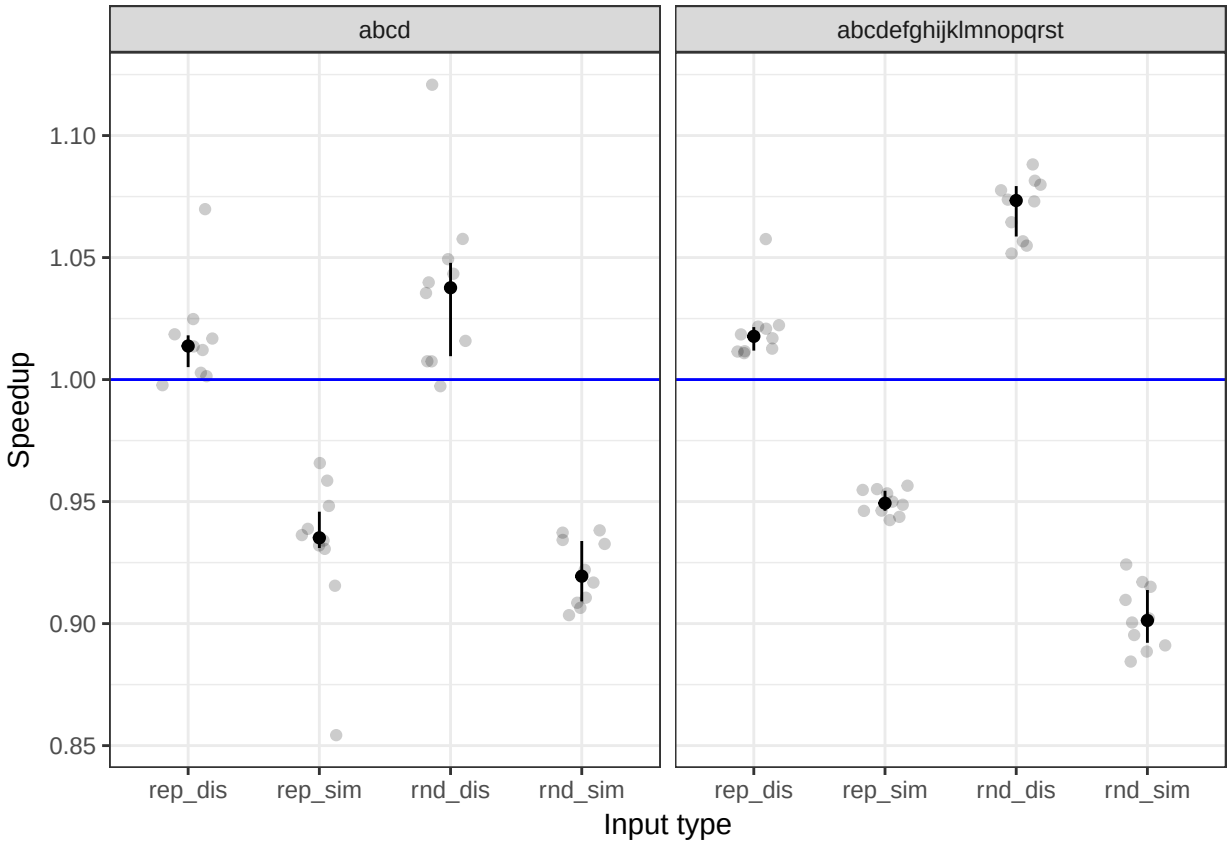


2.6 Maxrep

2.6.1 Vanilla Maxrep vs. non-maxrep



2.6.2 Rank&check Maxrep versus Vanilla Maxrep



3 Optimizations on parent operations

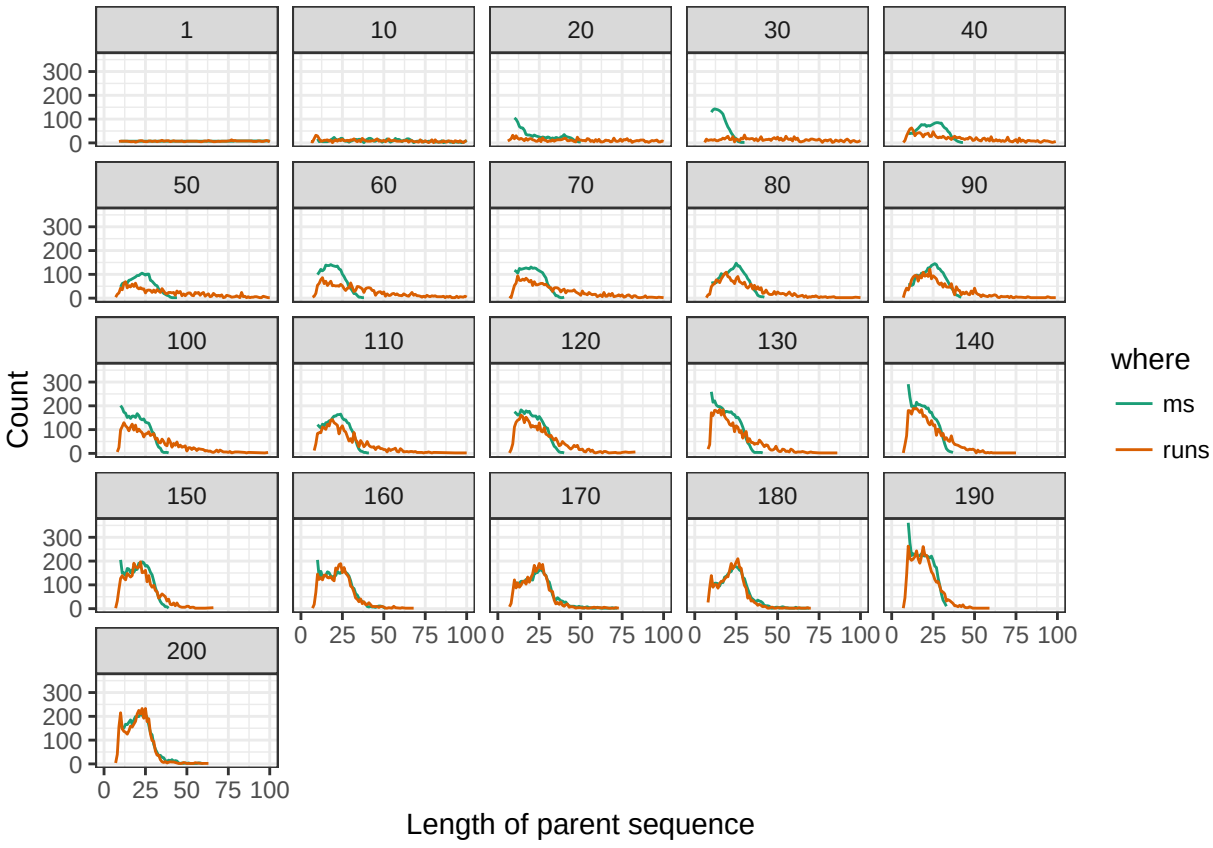
3.1 Input data

We generate the index input string with repetitions as follows. We generate a random seed block b of length 200. Next, we generate blocks of the same length b_k by introducing k mutations on b. The index string of length 10MB is $b \circ b_k^{(1)} \ldots \circ b_k^{(4999)}$.

The query string is obtained as a concatenation of labels from nodes of the suffix tree of s. We select nodes with node depth of at least 10 and string length at most 170 for a total string length of 103KB. We separate the labels with a sentinel character that does not appear in s.

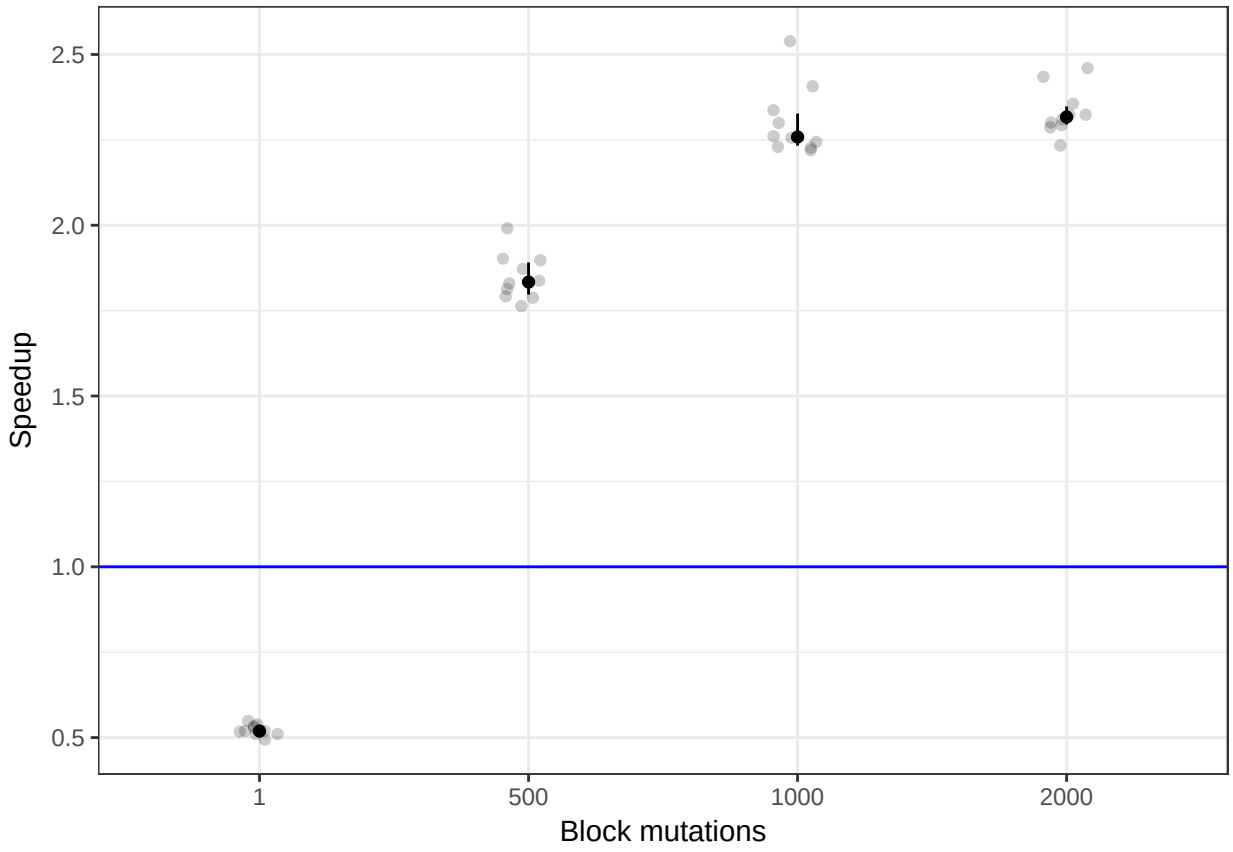
Furthermore, we perform experiments for various choices of $1 \geq k \geq |b|$.

The plot below shows a histogram of the length of consecutive parent operations. This quantity is important since the speedup of this optimization is proportional to the length of sequence of parent operations. Importantly, the optimization might not even be beneficial if the length of the sequence of parent operations is less than 3.



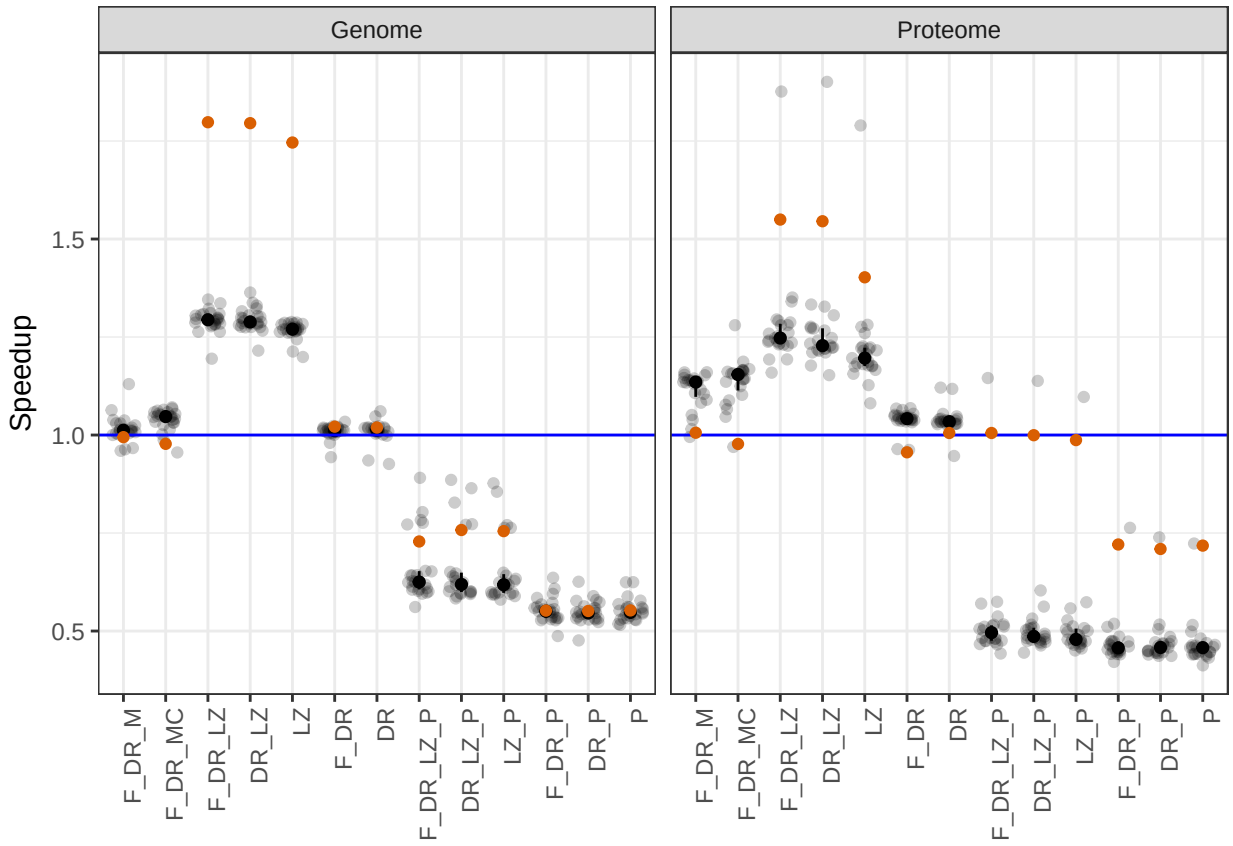
3.2 LCA versus parent sequence

```
## # A tibble: 40 x 5
## # Groups:   ntrial [10]
##    ntrial      k   lca  pseq     value
##     <int> <fctr> <int> <int>     <dbl>
##  1      1      1   887   459 0.5174746
##  2      1   1000   207   476 2.2995169
##  3      1   2000   213   487 2.2863850
##  4      1    500   225   408 1.8133333
##  5      2      1   863   448 0.5191194
##  6      2   1000   209   469 2.2440191
##  7      2   2000   216   502 2.3240741
##  8      2    500   234   430 1.8376068
##  9      3      1   861   459 0.5331010
## 10      3   1000   205   479 2.3365854
## # ... with 30 more rows
```

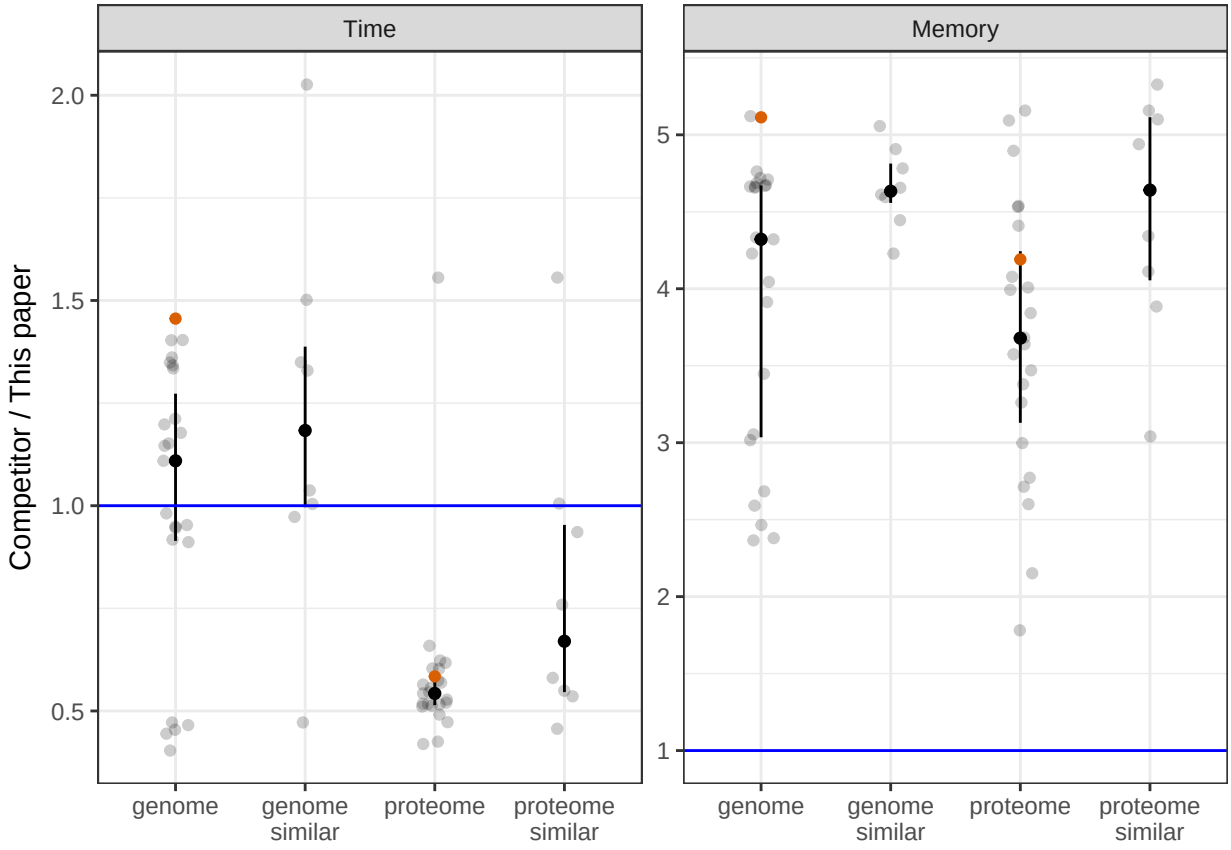

2.5
2.0
1.5
1.0
0.5
Speedup
1
500
1000
2000
Block mutations

4 Genome tests

4.1 Figure 4



4.2 Figure 5


Time
Memory
Competitor / This paper
2.0
1.5
1.0
0.5
5
4
3
2
1
genome
genome similar
proteome
proteome similar