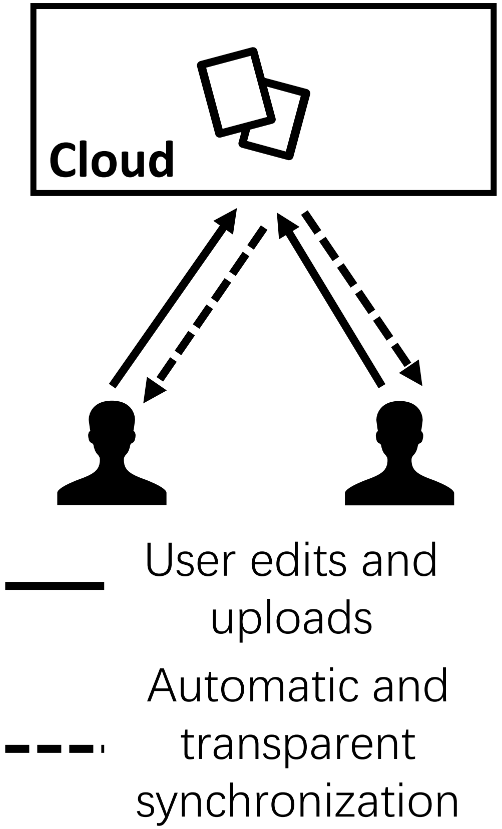
Lock-Free Collaboration Support for Cloud Storage Services with Operation Inference and Transformation

FAST’20

How to collaborate online

- ➢ Version Control Systems: Git, SVN…
  - Complex operations
  - Not suited to non-technical users
- ➢ Dedicated online editors: Google Docs, Overleaf…
  - Web-based
  - Easy-to-use
  - But limited functionality
- ➢ Cloud storage services: Dropbox, OneDrive…
  - Combines the advantages of the former
  - Transparent Synchronization



Cloud

User edits and uploads

Automatic and transparent synchronization

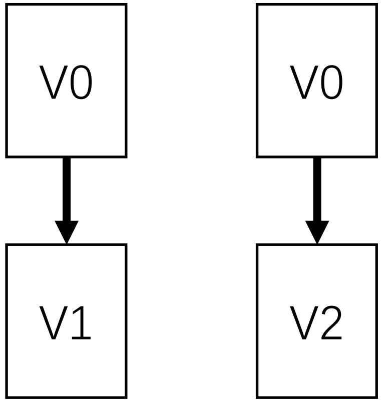
Cloud storage services

Conflict and existing resolution

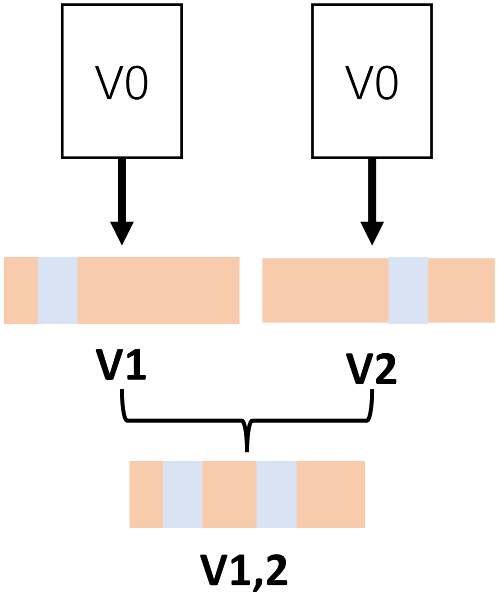
- Unavoidable conflicts
  - Conflicts occur when multiple users modify a file at the same time
- Use file-level locks to minimize conflicts
  - It is not worth using locks due to network latency
  - Passive locks(Nutstore etc.) may not accurately determine the time to lock or release
  - Active locks(BOX etc.) may cause others to enter the critical area

Many conflicts are not ”true”

- ➢ Most file-level conflicts can be resolved correctly



File-level conflict occurs



Actually possible to correctly merge

Main Idea

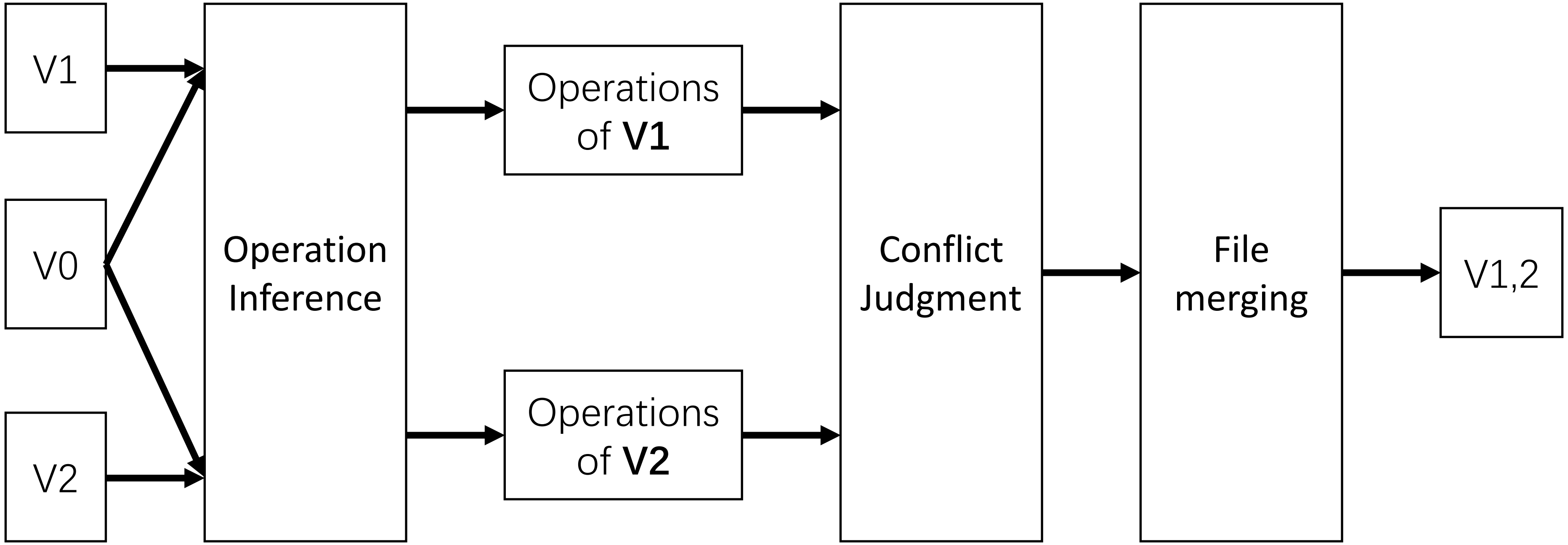
- ➢ Lock is inefficient

  -> No lock is used to achieve the transparency and user-friendliness

- ➢ Many conflicts are not ”true”

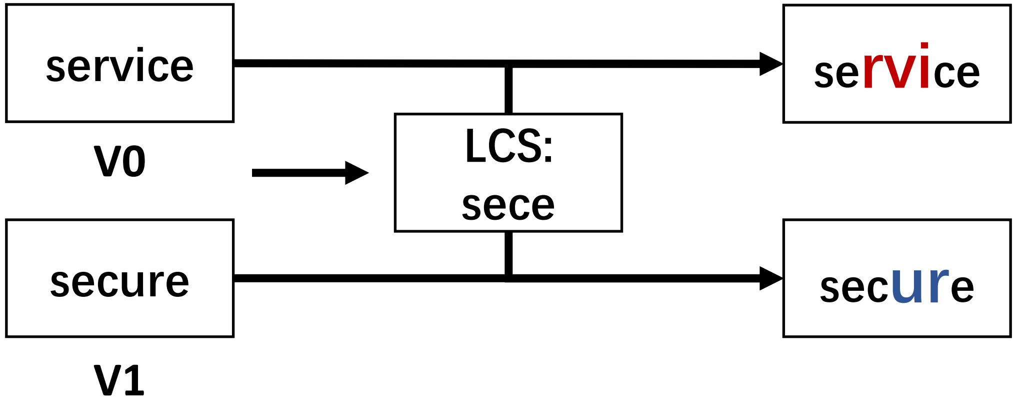
  -> Fine-grained inference of conflicting files to resolve conflicts

Architecture Overview



Operation Inference

- Unify edit operations as **“insert”** or **“delete”**
- Inferring operations by the longest common subsequence(LCS)



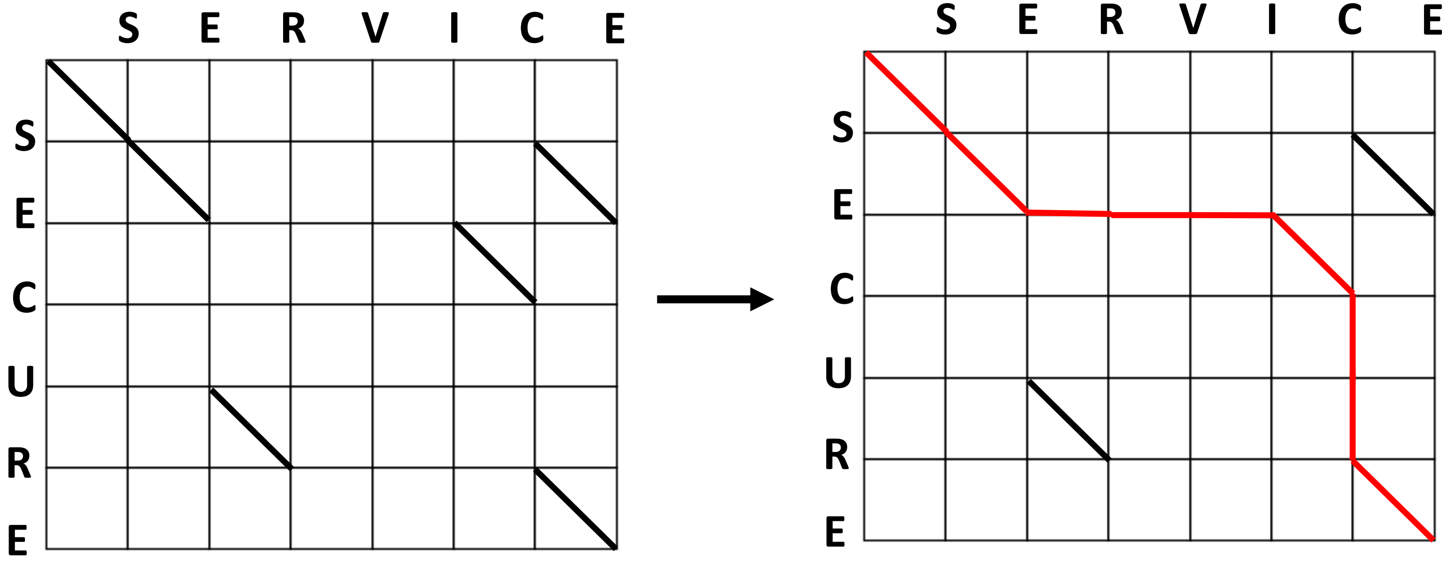
Operation Inference

Challenge:

Dynamic planning for LCS can take a lot of time: 30S for 500KB file

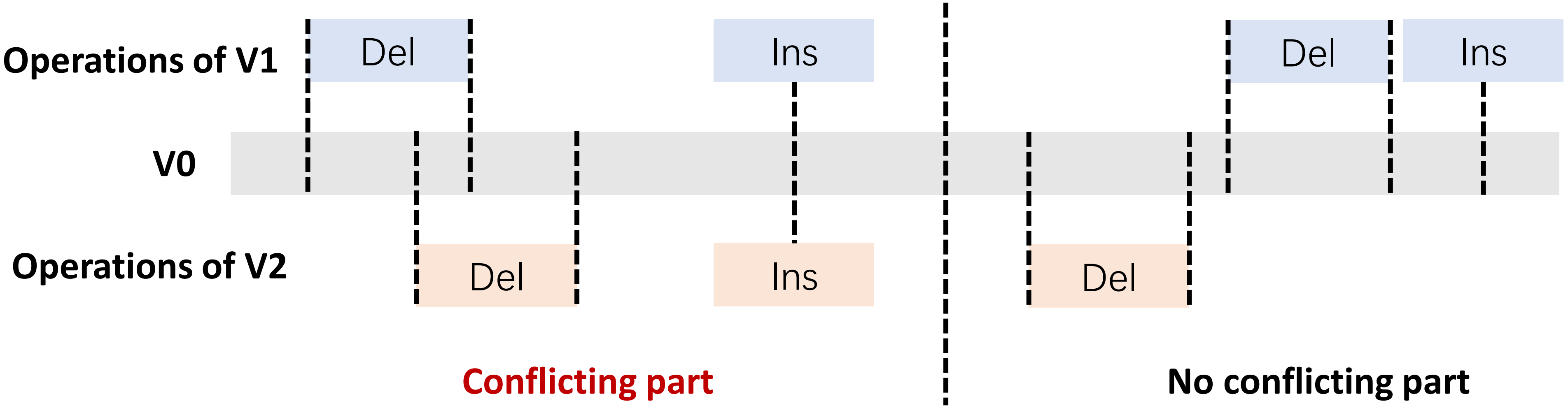
Resolution:

Use edit graph to calculate LCS, increasing speed by 150 times and achieving a nearly linear time overhead.



Operational Transformation

- Delineation of the impact region for each operation
  - The impact region of an insert operation is the position where insert occurs
  - The impact region of a delete operation is the entire deleted part
- If the impact region of two operations overlap, means that a conflict has occurred between the operations



Operational Transformation

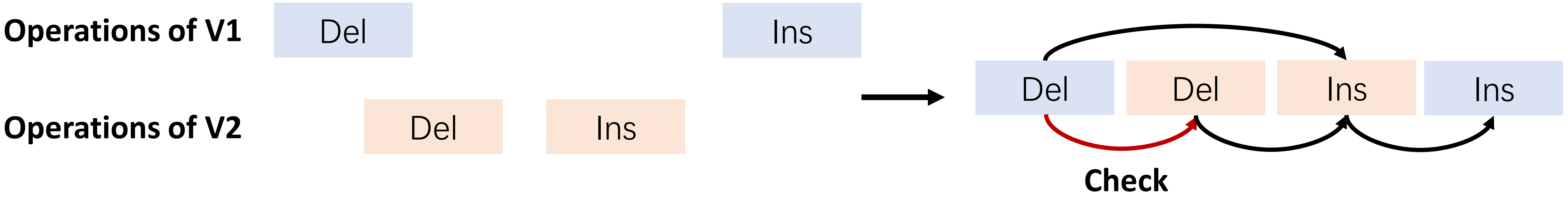
Challenge:

How to identify and handle conflicting operations as fast as possible?

Resolution:

For fast conflict detection, sort operations first and check backwards only

Enables linear time complexity due to the fact that conflicts rarely occur



Operational Transformation

Challenge:

How to preserve the user's intent intact while minimize conflicts?

Resolution:

Discuss the different types of conflicts separately

V1:
We need clothes, and books.

V0:
We need foods, water, clothes, and books.

V2:
We need foods, and books.

→

V1,2:
We need [Alice delete:foods,]
[Bob delete:clothes,]and books.

Del-Del

Operational Transformation

V1:
There is a cat.

V0:
There is a cat in the courtyard.

V2:
There is a cat in the spacious courtyard.

→

V1,2:
There is a cat[Alice delete: in the][Bob insert: spacious][Alice delete: courtyard].

Del-Ins

V1:
We need foods, water, and books.

V0:
We need foods and books.

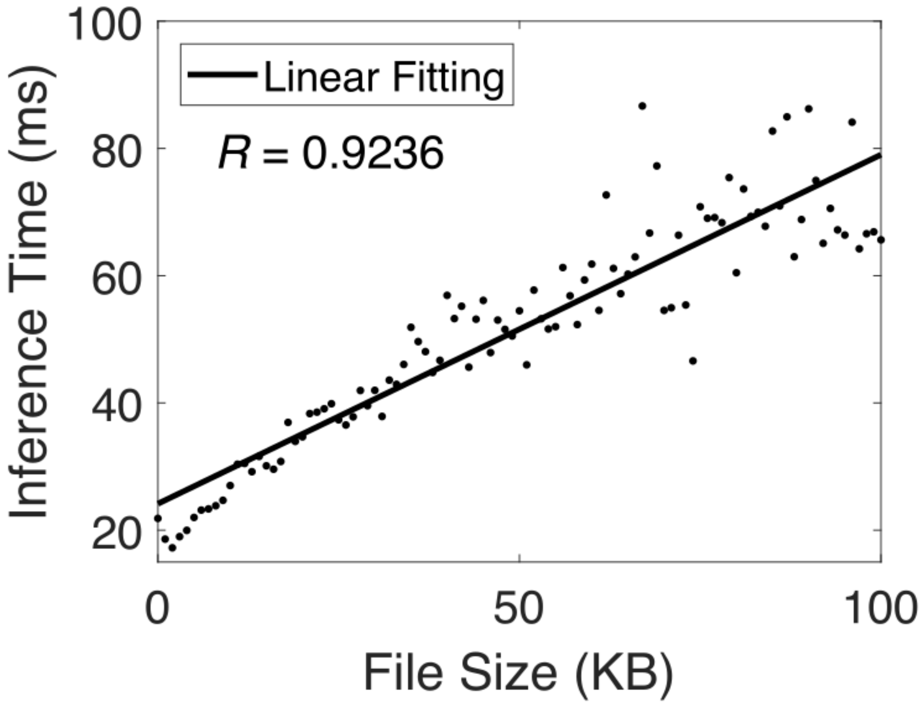
V2:
We need foods, clothes, and books.

→

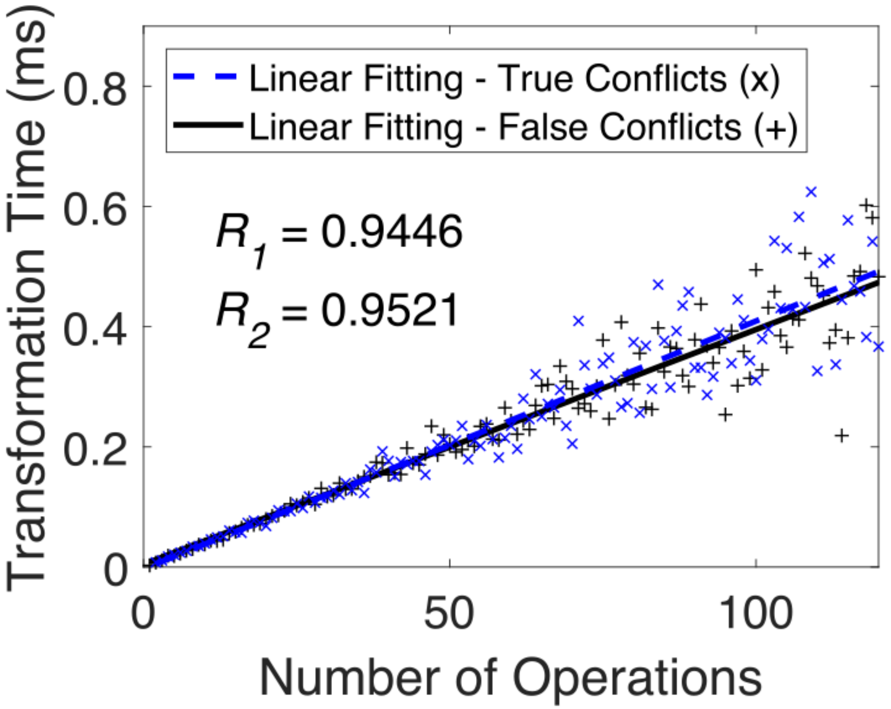
V1,2:
We need foods, [Alice insert:water] [Bob insert:clothes], and books.

Ins-Ins

Evaluation

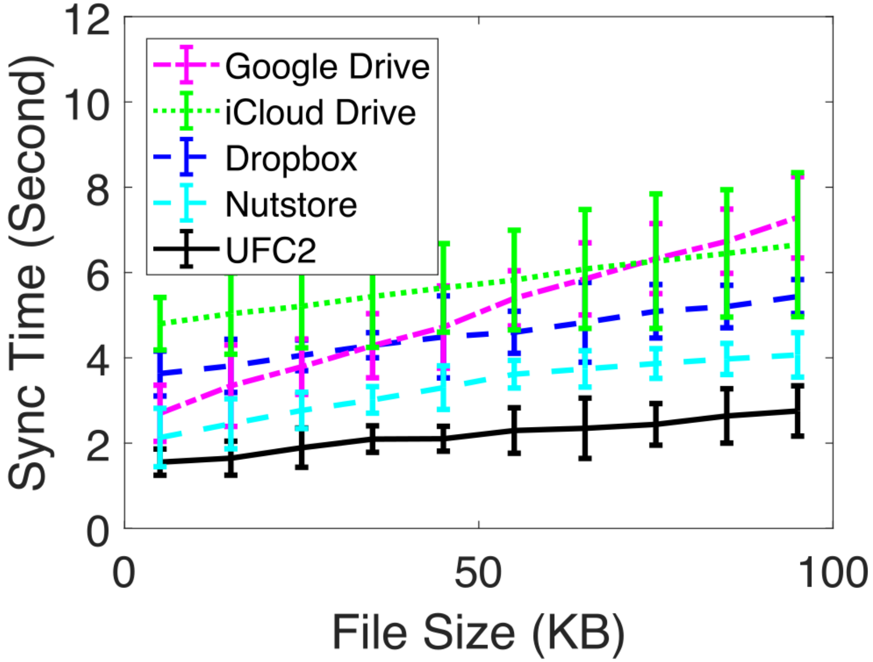


Time overhead incurred by devised operation inference

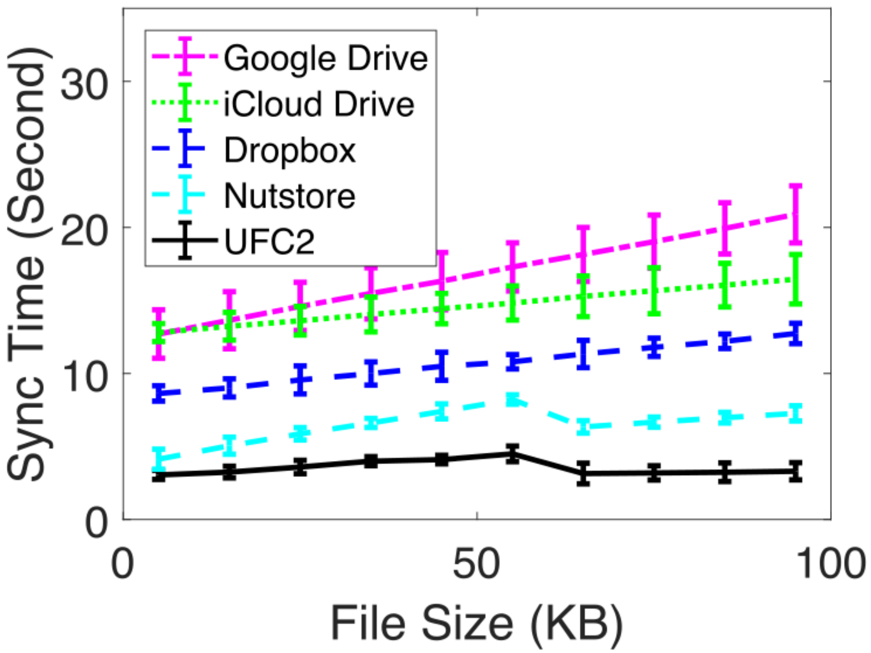


Time overhead incurred by devised operation transformation

Evaluation



Sync time of UFC2 and representative cloud storage services for a file update when there are no file-level conflicts



Sync time of UFC2 and representative cloud storage services for a file update when there exist file-level conflicts