# On Selecting the Right Optimizations for Virtual Machine Migration

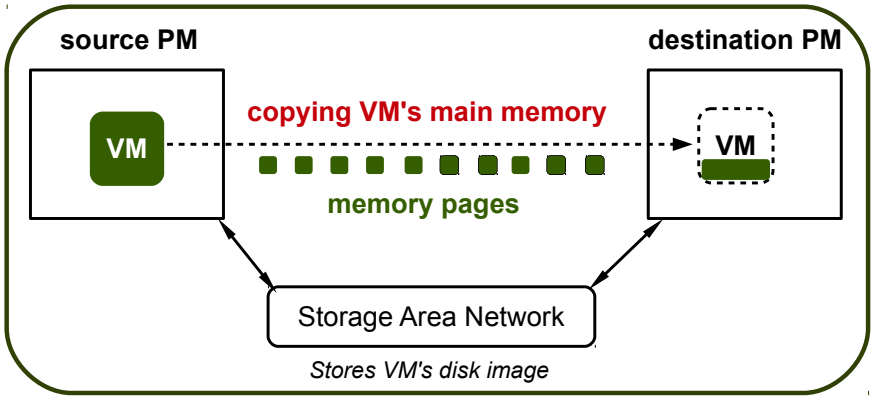
**Senthil Nathan N**, Umesh Bellur, Purushottam Kulkarni

Systems and Networks Group (SynerG)
Department of Computer Science and Engineering
Indian Institute of Technology Bombay

April 2, 2016

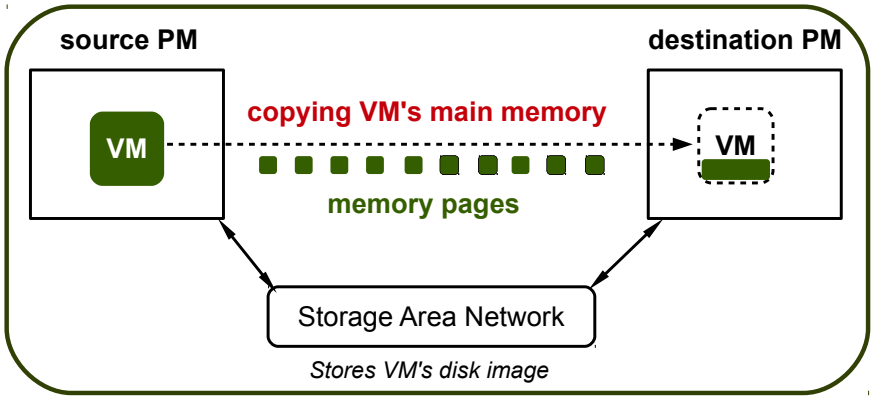
Virtual Execution Environments (VEE) 2016

# Pre-Copy Live Migration

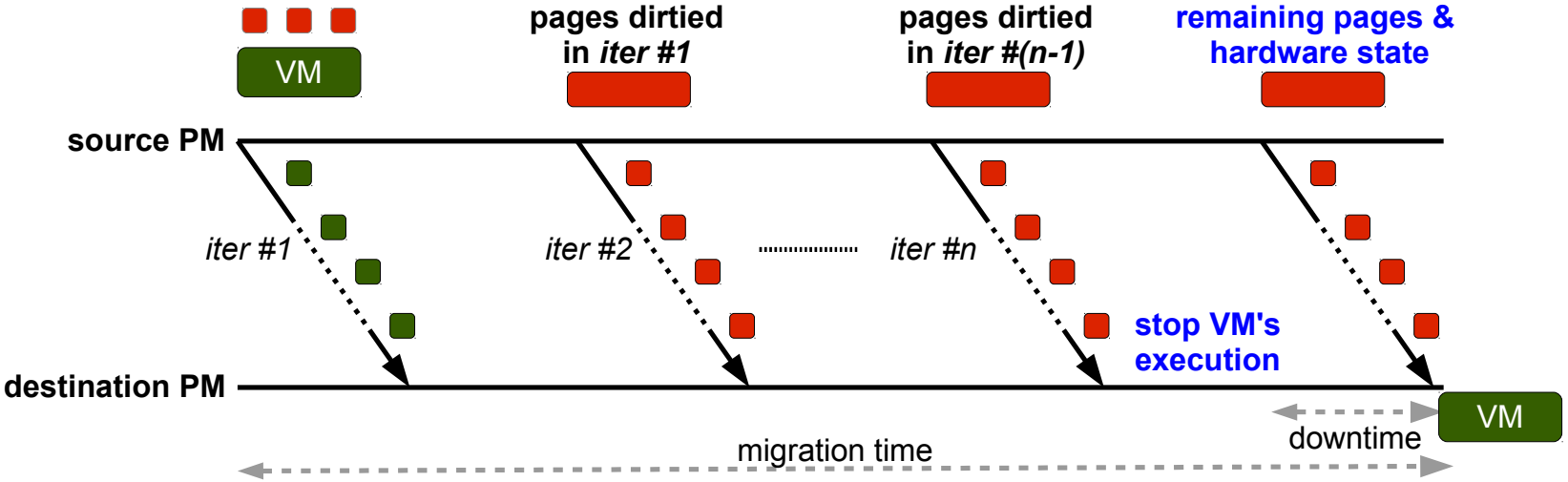



VM is executing: Memory pages are transferred iteratively

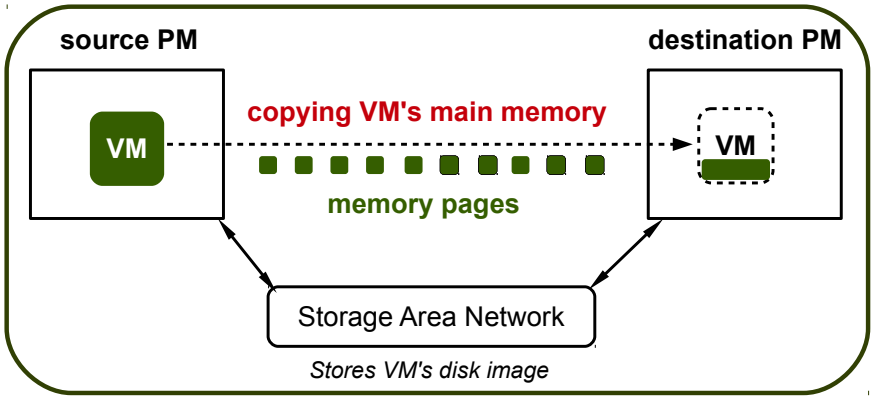
# Pre-Copy Live Migration



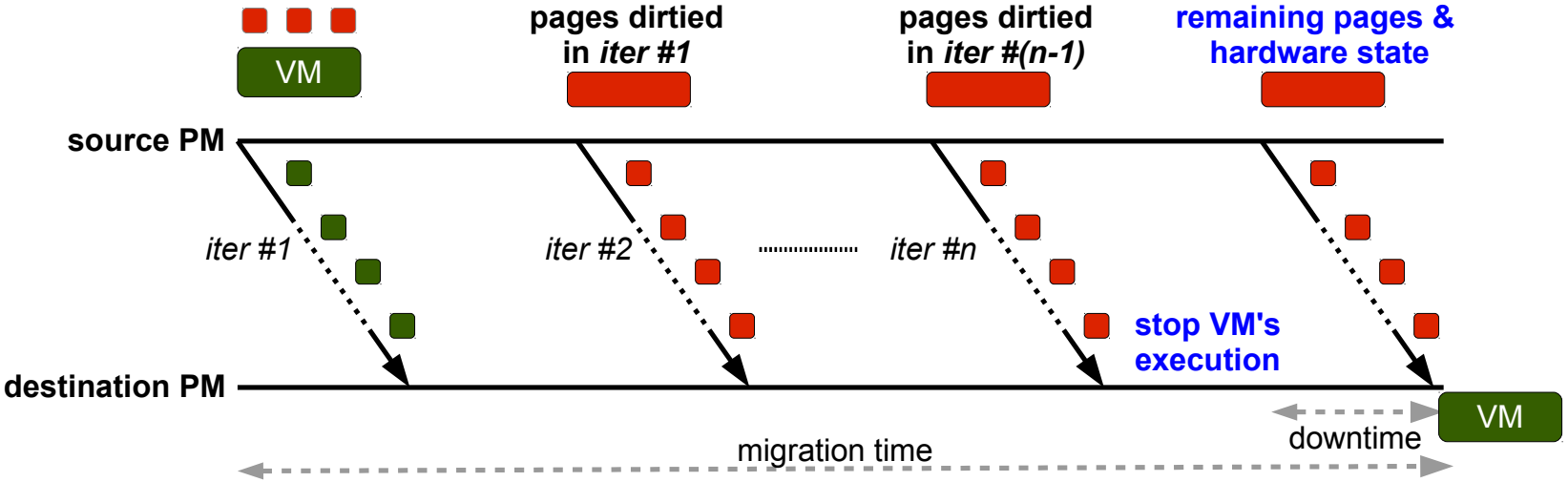

VM is executing: Memory pages are transferred iteratively



# Pre-Copy Live Migration




VM is executing: Memory pages are transferred iteratively




migration time(VM hosting file server) = 5 minutes

# Existing Optimization Components for Pre-Copy Live Migration

## Page Dirty Characteristics

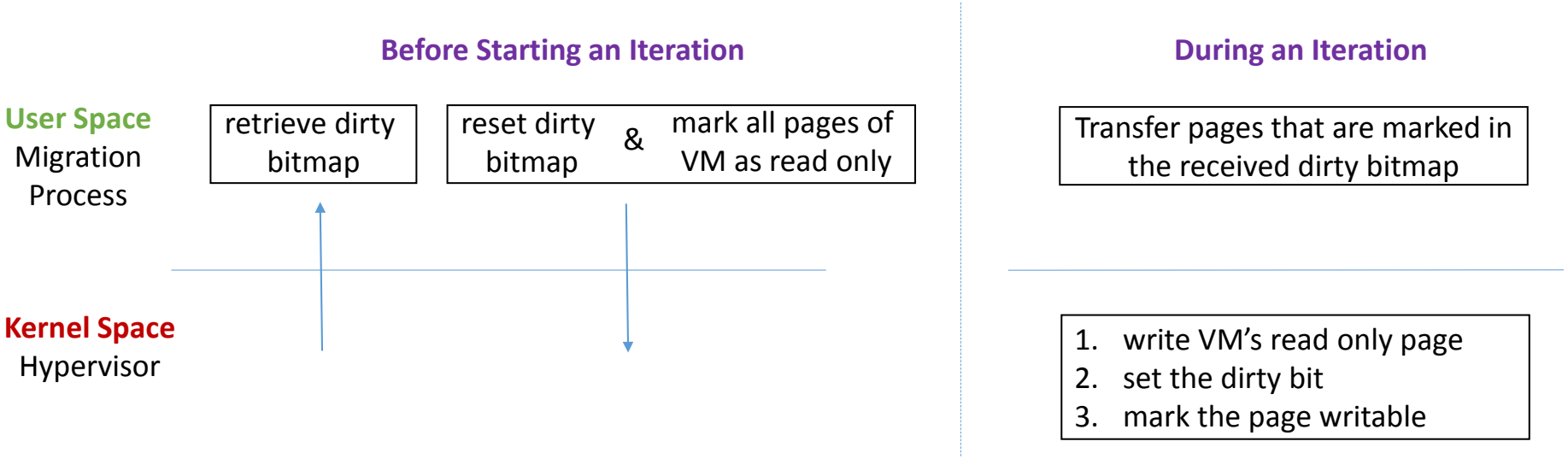
1. page delta (del).
2. frequently/hot dirty pages (hdp).

## Page Content Characteristics

1. zero pages (zp) & sub zero pages (szp).
2. duplicate pages (dup) & sub duplicate pages (sdup).
3. word-level duplicates (wld).

We identifed a new component: **False Dirty Page** (fdp)

# False Dirty Pages (fdp)




1. An iteration can transfer a page that is already dirted during current iteration.
2. What happens if this page never get dirtied again before the next iteration?
   - An unmodified page is transferred: False Dirty Page.

# Existing Optimizations

| Techniques | Account for | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | *dirty characteristics* | | | *content characteristics* | | | | |
| | fdp | del | hdp | zp | szp | dup | sdup | wld |
| Delta Compression | ✓ | ✓ | | | | | | |
| Page Skip | ✓ | | ✓ | | | | | |
| Full Page Deduplication | ✓ | | | ✓ | | ✓ | | |
| Sub Page Deduplication | ✓ | ✓ | | ✓ | ✓ | ✓ | ✓ | |
| Data Compression | | | | ✓ | ✓ | | | ✓ |

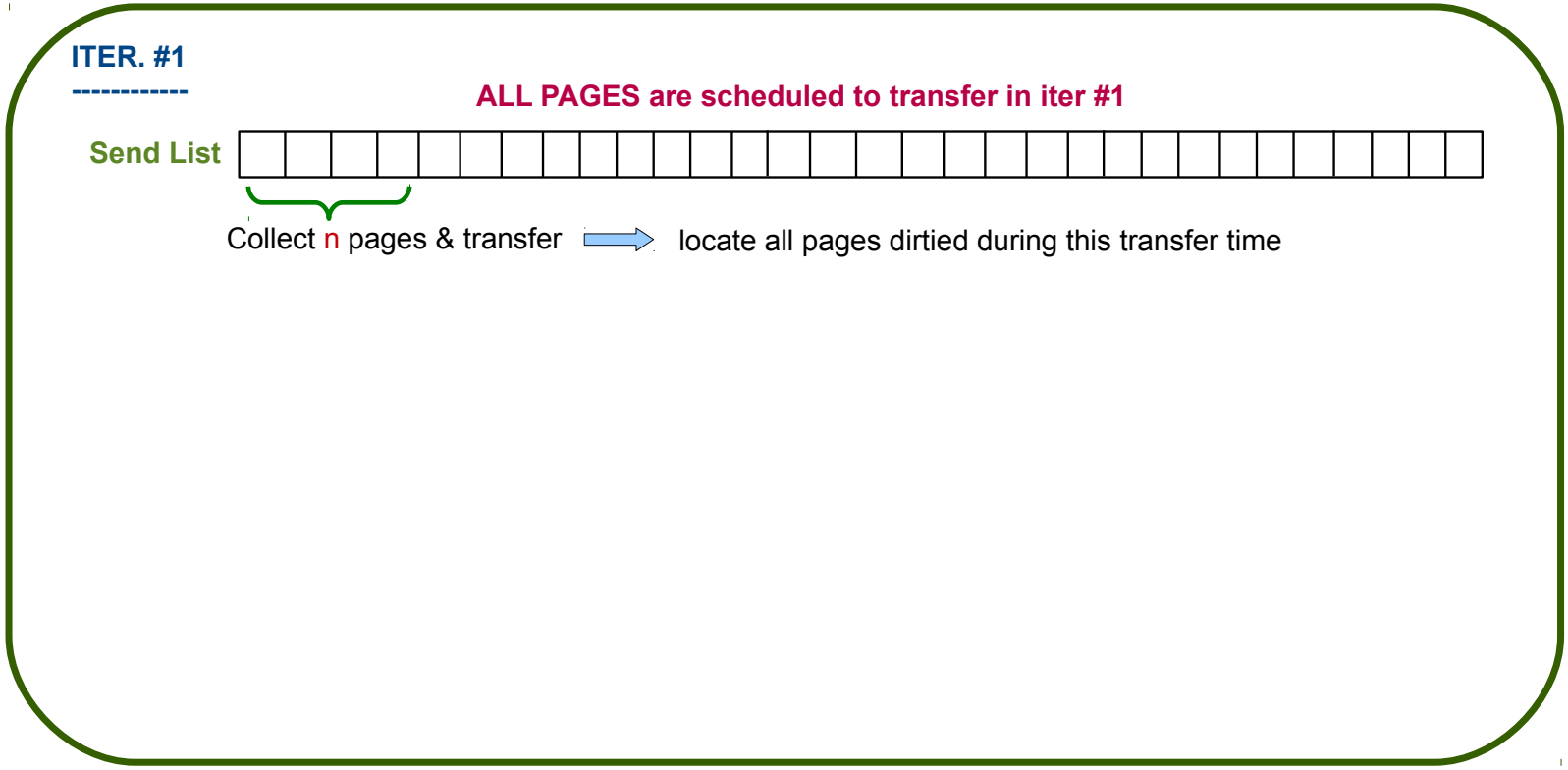
**False Dirty Page** (fdp) is overlooked in existing study

## Question of Interest

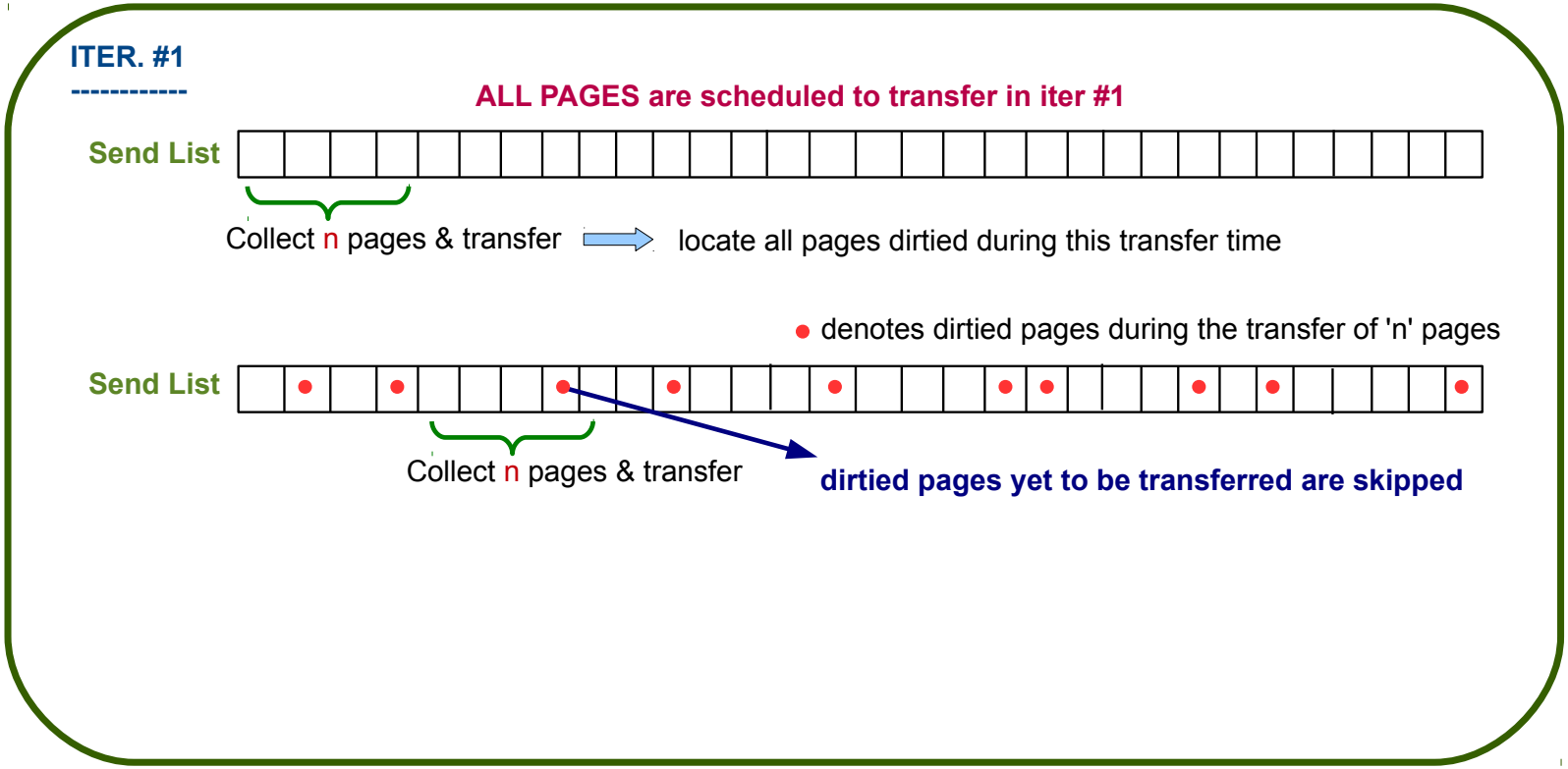
1. Which is the best technique in terms of performance & cost?
2. Does optimization's performance depend on application behaviour?
3. What combinations of optimizations are more appropriate?
4. Can the combinations of optimizations reduce the resource cost?

fdp **helps in answering these questions**

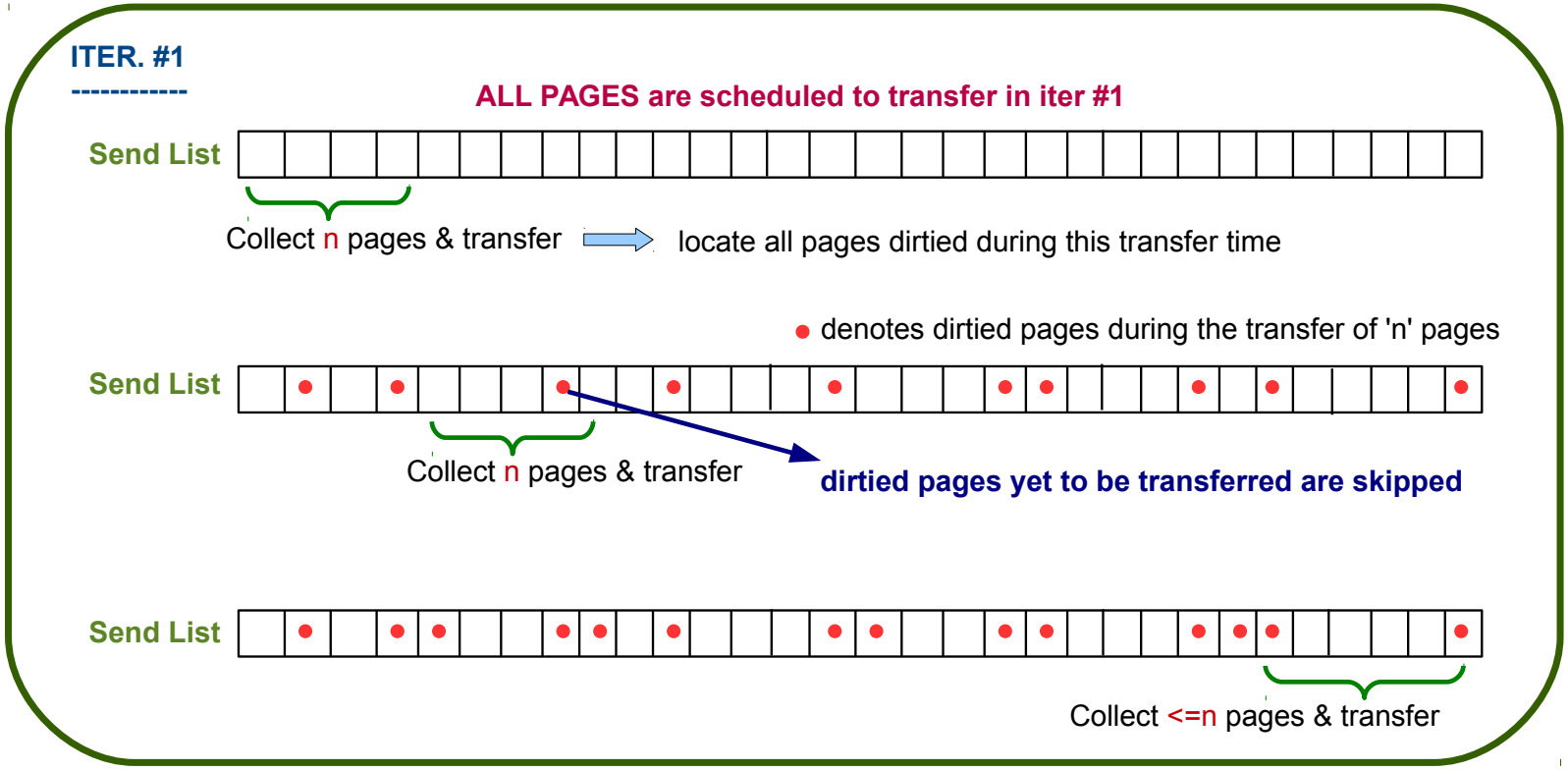
# Page Skip

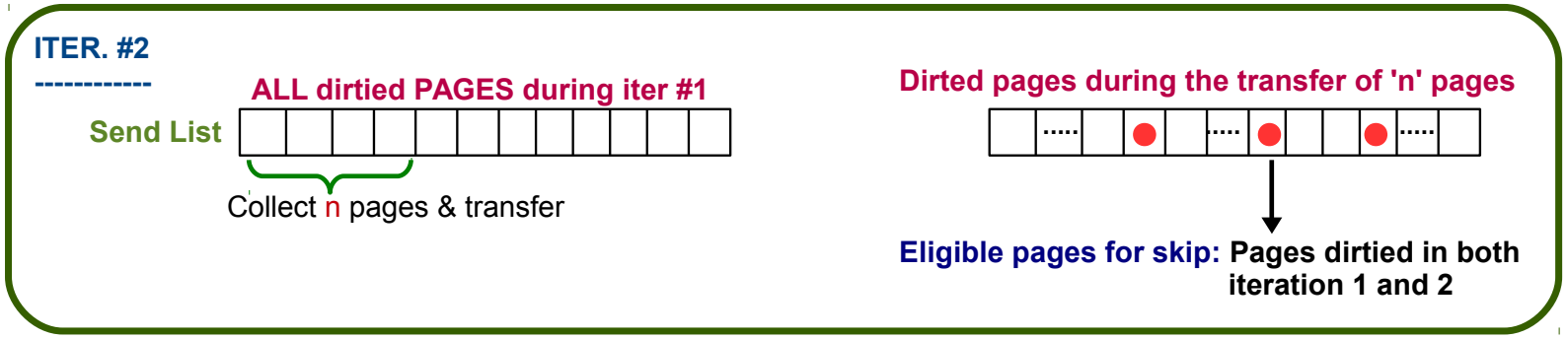

ITER. #1
---
ALL PAGES are scheduled to transfer in iter #1
Send List
Collect n pages & transfer
locate all pages dirtied during this transfer time

# Page Skip


ITER. #1
---
ALL PAGES are scheduled to transfer in iter #1
Send List
Collect n pages & transfer
locate all pages dirtied during this transfer time
denotes dirtied pages during the transfer of 'n' pages
Send List
Collect n pages & transfer
dirtied pages yet to be transferred are skipped

# Page Skip


ITER. #1
ALL PAGES are scheduled to transfer in iter #1
Send List
Collect n pages & transfer
locate all pages dirtied during this transfer time
denotes dirtied pages during the transfer of 'n' pages
Send List
Collect n pages & transfer
dirtied pages yet to be transferred are skipped
Send List
Collect <=n pages & transfer


ITER. #2
ALL dirtied PAGES during iter #1
Send List
Collect n pages & transfer
Dirted pages during the transfer of 'n' pages
Eligible pages for skip: Pages dirtied in both iteration 1 and 2

# Our 3 Goals

1. To perform a comprehensive empirical study to understand and compare between performance and cost of different migration optimization techniques.

2. To understand the relationship between application behavior and the payback of different optimizations.

3. To investigate the feasibility of combining optimizations so as to get the maximum improvement in migration performance for unit resource cost.

# Experimental Setup and Workloads Used for Evaluation

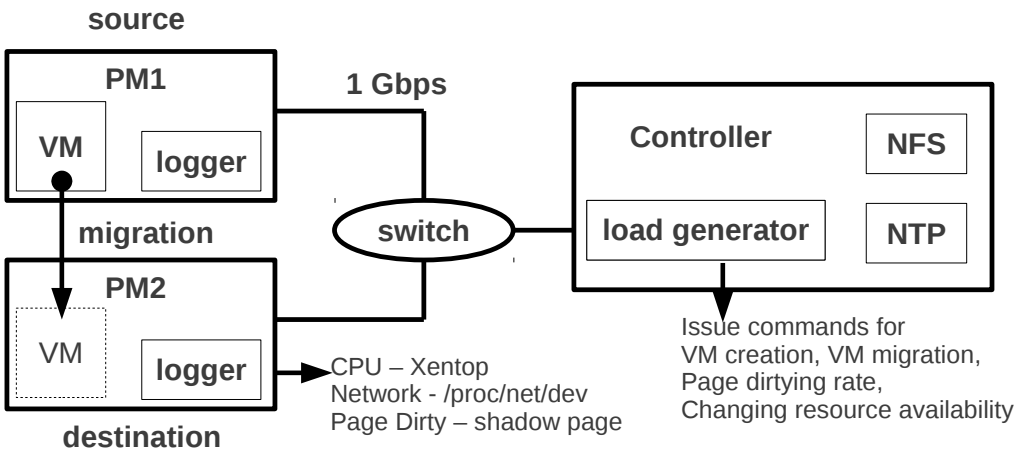



Table : Workloads: Web Services

| Workloads | Remark |
|---|---|
| Mediawiki, DVD Store, RUBiS | webserver and database server |
| OLTPBench | *epinions, twitter, seats, voter, tpcc, tatp, ycsb* |
| HTTP file server | 100s of files each with different sizes |

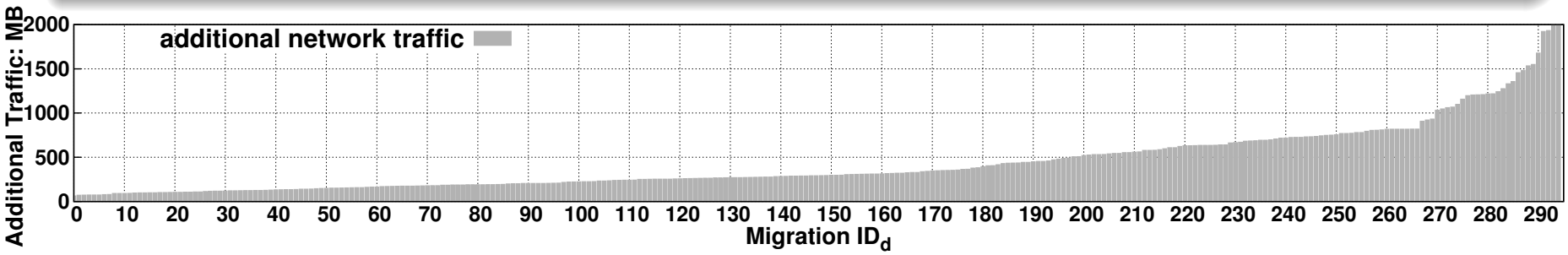
42 applications, 7 different migration rates.
$42 \times 7 = 294$ measured values.

Table : Workloads: Multimedia, Biotech, Mining, MPI

| Workloads | Remark |
|---|---|
| gcc | kernel compile with multiple threads |
| File compression | compress 8 GB data set with tar |
| Linpack | linear equation solver |
| Nu-Minebench (data mining) | *ECLAT, HOP, ScalParC, UtilityMine* |
| Mummer | gnome sequence alignment |
| Parsec (multimedia tools) | *bodytrack, ferret, fluidanimate, freqmine, vips, and x264* |
| Dacapo (java benchmarks) | *avrora, eclipse, fop, h2, jython, luindex, lusearch, pmd, sunflow, tomcat, tradebeans, tradesoap, and xalan* |

# Expectations on Optimization Performance

Performance of Delta Compression & Page Skip ∝ Additional Network Traffic Generated




Performance of Deduplication ∝ Additional Network Traffic & the Amount of Zero, Duplicate Pages

Performance of Data Compression ∝ Total Network Traffic & the Amount of Zero, Word Level Dup.

# Methodology

1. Apply each optimization individually on 294 migration instances.

2. If expectation fails, identify the reason (i.e., application behavior).

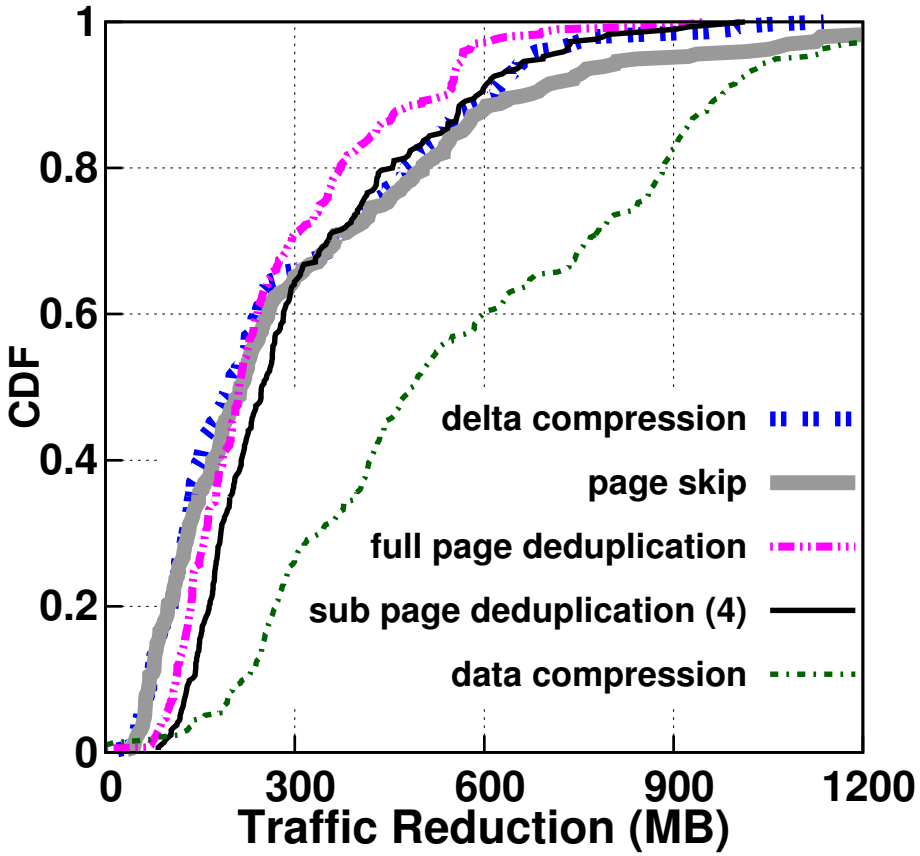
3. Compare performance & cost of each optimization and find appropriate combinations.

# Goal 1: Performance and Cost Trade-Off

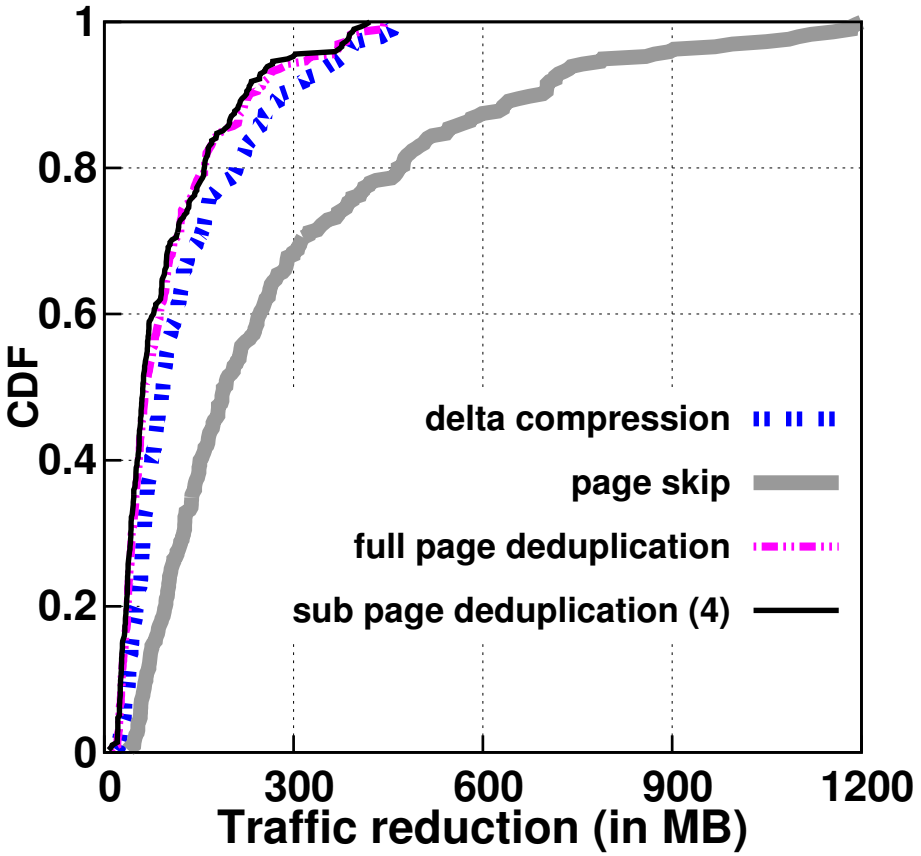
| **Techniques** | **Avg. traffic reduction** | **Increase in CPU at** | |
|---|---|---|---|
| | | source PM | destination PM |
| Page Skip | 20 % | **1 x** | **1 x** |
| Data Compression | **37 %** | 5 x | 1.3 x |
| Delta Compression | 17 % | 3.2 x | 1.2 x |
| Full Page Dedup. | 17 % | **11 x** | 1 x |
| Sub Page Dedup. | 20 % | **13 x** | 1.2 x |

**Page skip is the best optimization in terms of performance-cost tradeoff**

# Significant Traffic Reduction due to fdp & hdp




(a) Total traffic reduction




(b) Traffic reduction (fdp+hdp)

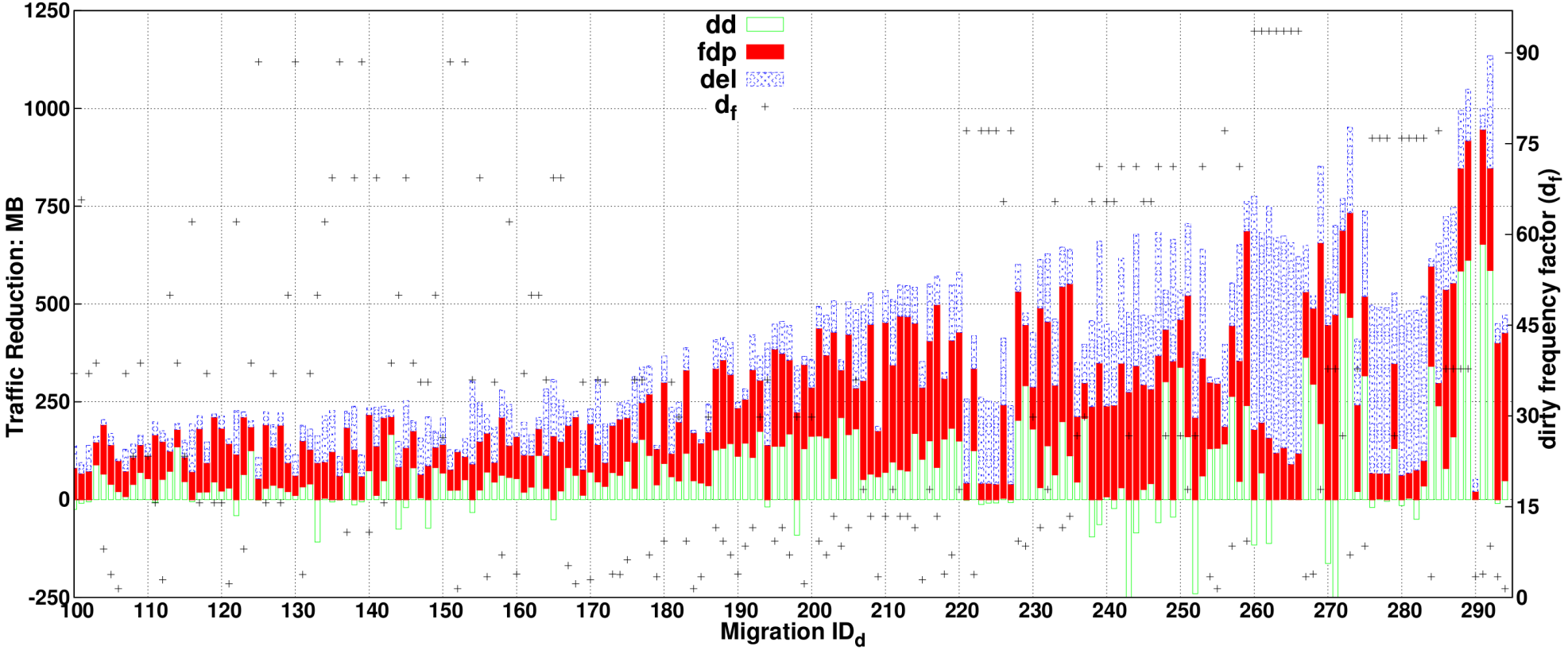
## Observations: Reduction in network traffic

- Reduction due to data compression is significant.
- On average, 40% reduction due to False Dirty Pages (fdp).
- Approximately, 53% reduction due to Hot Dirty Pages (hdp).

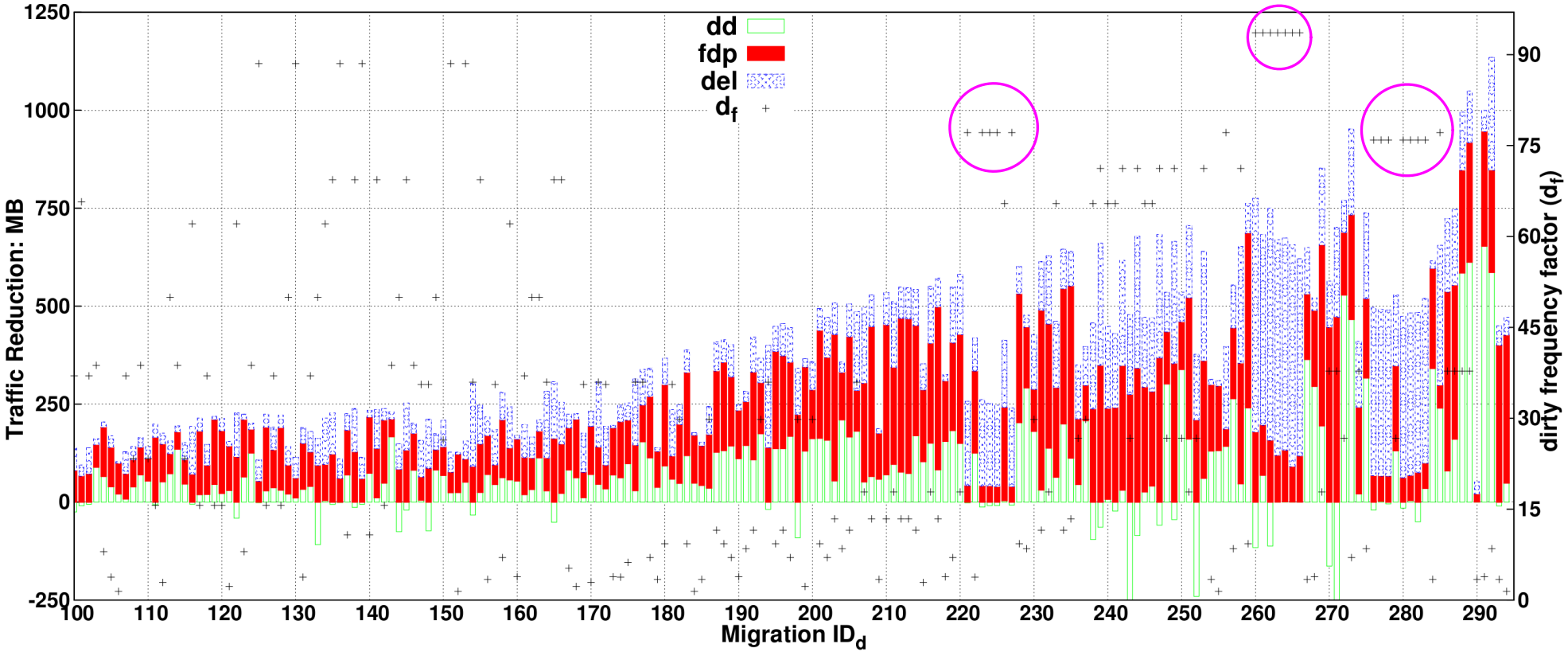
# Goal 2: Co-relation Between Application Behavior and Optimization's Performance

| Techniques | Low performance when |
|---|---|
| *Delta Compression* | high dirty frequency, high disk read, `memset()`,`free()` |
| *Page Skip* | page dirty rate $<<$ migration rate and large writable set size |
| *Deduplication* | high dirty frequency and low `dup+zp` |
| *Data Compression* | low word level duplicates & zero content |

# Delta Compression: Impact of Application Behaviour



# Delta Compression: Impact of Application Behaviour



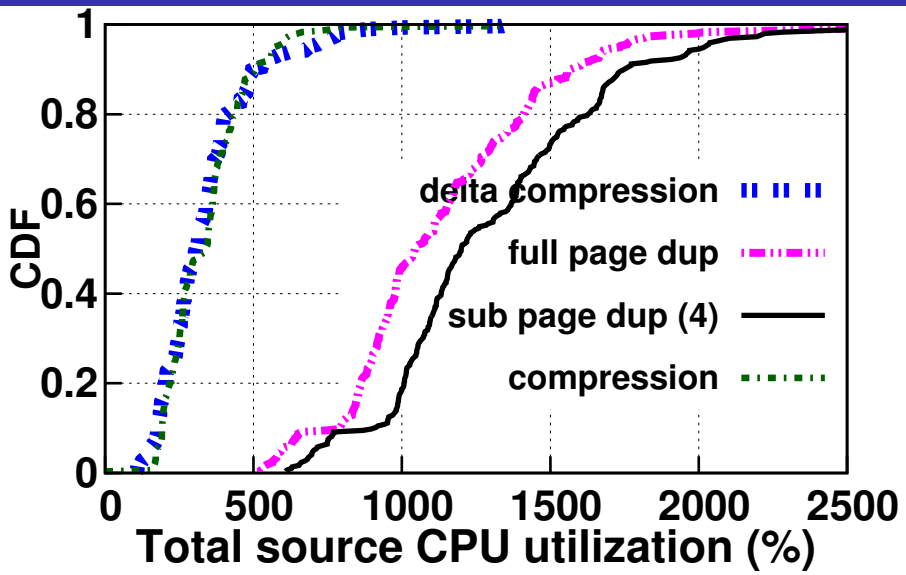

- fdp $\propto \frac{1}{d_f}$
- high $d_f \implies$ improvement depedentent of page modification size.
  - `disk read`, `memset()`, `free()` modified all 4 KB.

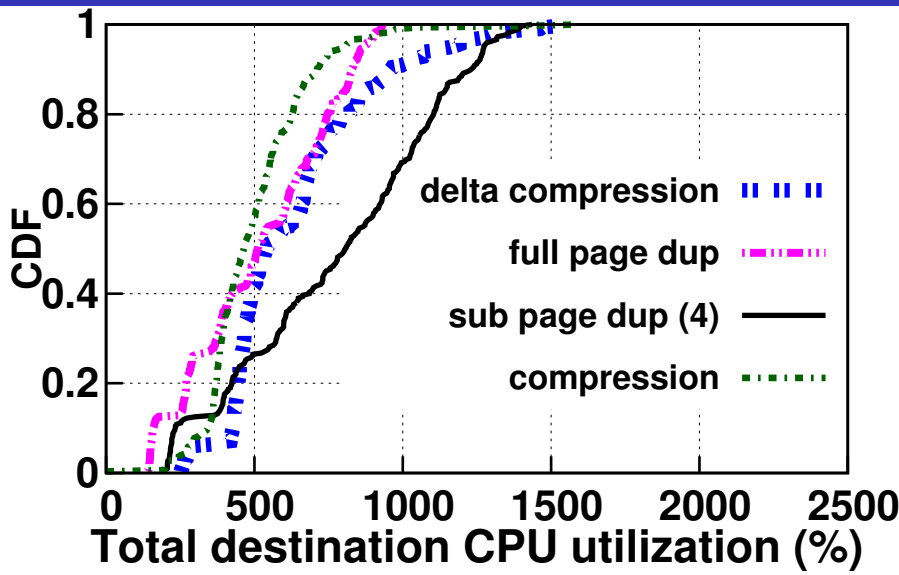
# Goal 3: Appropriate Combinations—Page Skip as Base Optimization

| **Page Skip +** | **Reduction in traffic** (over skip) | **Total Increase in CPU at** | |
|---|---|---|---|
| | | source PM | destination PM |
| Data Compression | **21 %** | 4.7 x (-.3) | 1.3 x |
| Delta Compression | 1.4 % | 2.7 x (-.5) | 1 x |
| Full Page Dedup. | 7 % | **10.8 x** (-.2) | 1 x |
| Sub Page Dedup. | 9 % | **13 x** | 1.2 x |

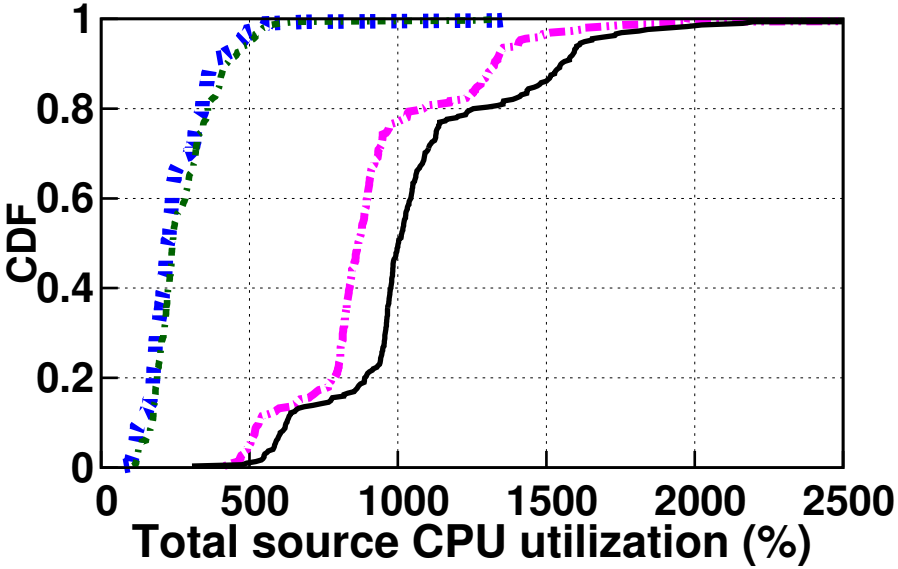
**Page Skip + Data Compression is the best combination in terms of performance-cost tradeoff**

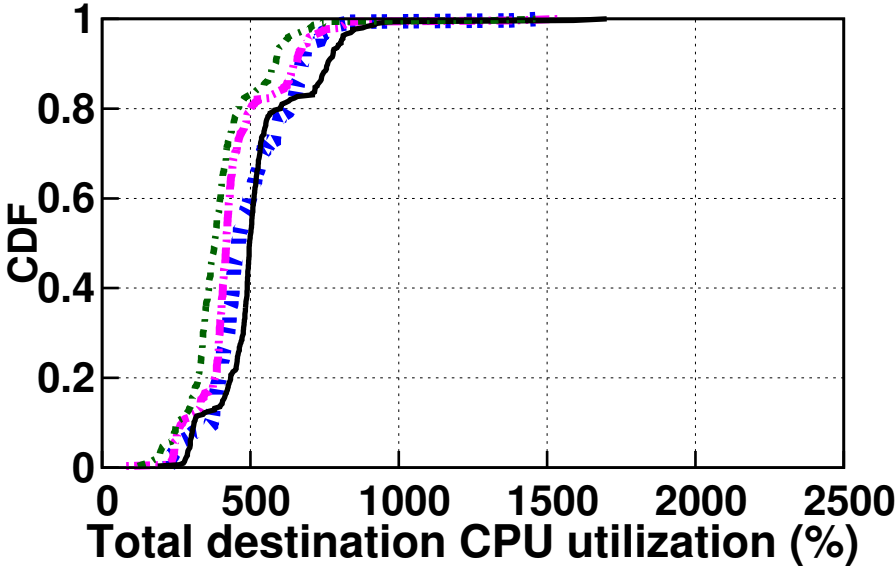
# Reduction in Total CPU Utilization due to Combination of Techniques



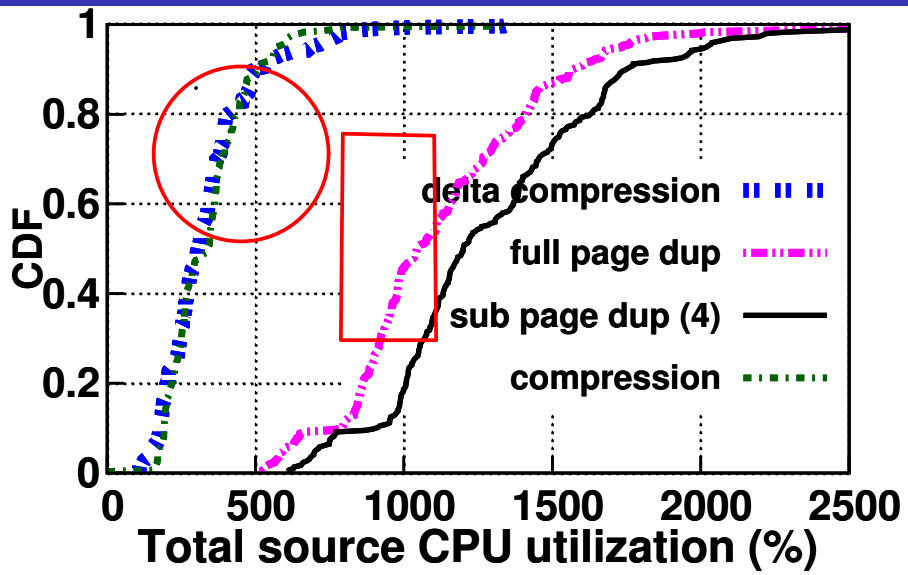

(a) Individual Technique

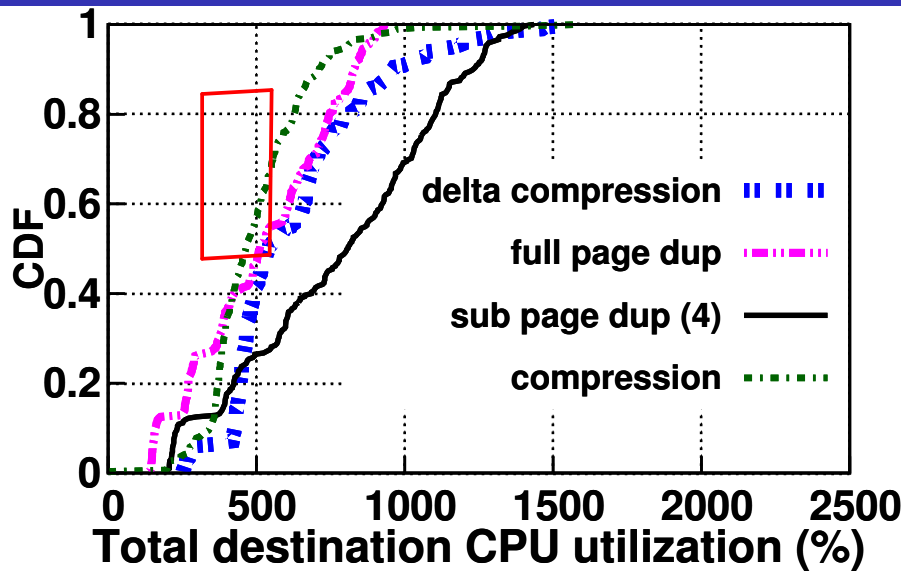


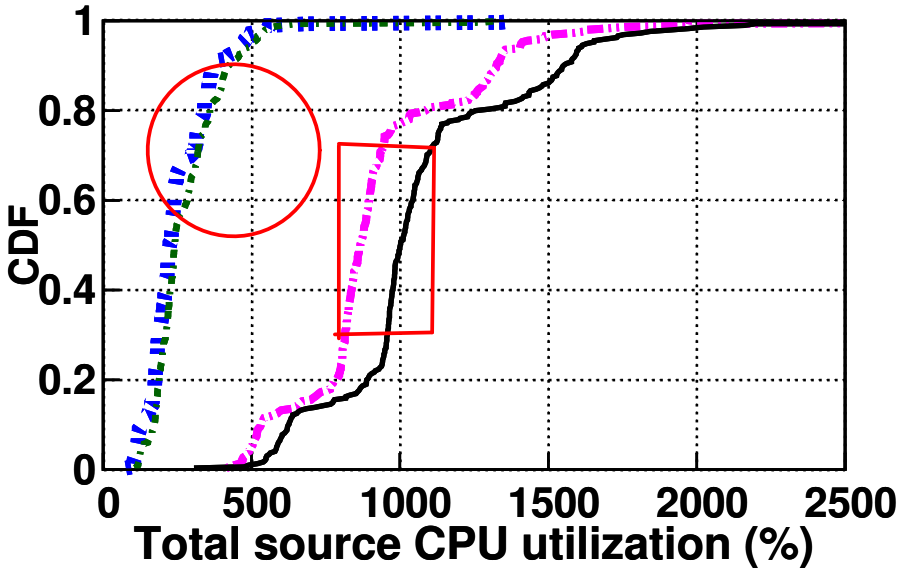

b) Page skip as the base technique

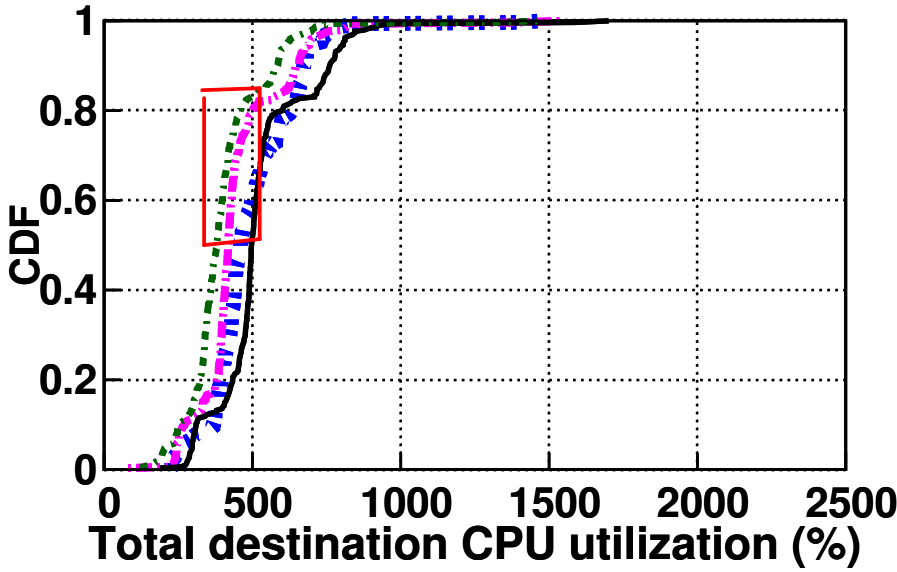
# Reduction in Total CPU Utilization due to Combination of Techniques




(a) Individual Technique




b) Page skip as the base technique

# Reduction in Impact of Application Behaviour due to Combination of Techniques

| Page Skip + | **low performance gain when** |
|---|---|
| *Delta Compression* | ~~high dirty frequency~~, unskipped pages are dirtied due to disk read, `memset(),free()` |
| *Deduplication* | ~~high dirty frequency~~, low `dup+zp` |
| *Data Compression* | low word level duplicates and low zero content |

# Appropriate Combinations—Page Skip + Data Compression as Base Optimizations

| **Page Skip + Data Compression +** | **Reduction in traffic** over page skip + data compression | **Total Increase in CPU at** | |
|---|---|---|---|
| | | source PM | destination PM |
| Delta Compression | 3 % | 7.1 x | 1.4 x |
| Full Page Dedup. | 2 % | **16 x** | 1.5 x |
| Sub Page Dedup. | **-3 %** | **18 x** | 1.5 x |

**No more suitable combinations**

# More Results (refer to paper)

1. Reduction in migration time and downtime due to each technique.
2. Detailed CPU utilization with respect to migration rate.
3. Impact of optimization parameters on performance improvement.
   - `cache size` for delta compression.
   - `build frequency` for page skip.
   - `sub page size` for deduplication.
4. Proposed a new cache replacement policy for delta compression that improves the performance over 30%.

# Summary of Key Results

1. Page Skip is the best optimization (must apply).

2. Page Skip + Data Compression is the best combination.

3. Deduplication technique need to applied with utmost care (can use KSM).

4. Combination of techniques reduce the
   - impact of application behaviour on optimization's performance.
   - total resource utilization (while increasing per second utilization).

# Thank You

We thank Google & Microsoft Research India for travel grant

# Questions?