# Push if you cannot Pull!

## Air-Caching and Media-on-Demand

Amotz Bar-Noy

| | |
|---|---|
| Past: | Tel Aviv University Israel |
| Present: | AT&T Research |
| Future: | Brooklyn College - CUNY |

# Windows Scheduling

Input:

- ⋆ The numbers $1, 2, \ldots, n$.
- ⋆ $W = \{1 \le w_1 \le w_2 \le \cdots \le w_n\}$: a set of integers. ($w_i$ is the window of $i$)
- ⋆ $h \ge 1$: an integer.

The problem:

- ⋆ Is it possible to schedule the numbers $1, 2, \ldots, n$ in a matrix with $h$ rows and infinite number of columns, such that occurrences of $i$ must appear no more than $w_i$ columns apart?
- ⋆ Construct a $h \times \infty$ matrix on the numbers $1, 2, \ldots, n$, such that the number of consecutive columns without $i$ is no more than $w_i - 1$.

# The Matrix

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\dots$ | $i$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $i$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\dots$ |
| $\dots$ | $j$ | $i$ | $\cdot$ | $\cdot$ | $k$ | $j$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $j$ | $\dots$ |
| $\dots$ | $k$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $k$ | $\cdot$ | $\cdot$ | $\cdot$ | $i$ | $\cdot$ | $\dots$ |
| $\dots$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\dots$ |

In the above schedule $h = 4$:
— The window of $i$ is at least $4$
— The window of $j$ is at least $5$
— The window of $k$ is at least $6$

Properties of "most" schedules:

⋆ Each row is represented by a cyclic schedule:
$\Rightarrow$ The matrix is cyclic (finite).

⋆ Each number appears only in one row.

⋆ Each row is a perfect schedule: The gap between $2$ appearances of $x$ is exactly $w'_x \leq w_x$ rows.

## Bounds

Optimization goal:

- Find the minimal $h$ with a positive answer.
- Denote such a bound: $H(W)$.

Lower bound:

- $h(W) = \sum_{i=1}^{n} \frac{1}{w_i}$.
- Because $i$ requires at least $1/w_i$ of a row.

Complexity:

- An NP-Hard problem even for $h = 1$.
- Approximation goals:
  - Minimize $H(w) - h(W)$.
  - Minimize $H(w)/h(W)$.

# Examples

| | | | | | |
|---|---|---|---|---|---|
| 1 | 1 | 2 | 1 | 1 | 2 |

- An imperfect solution for $W = \{2, 3\}$ and $h = 1$.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 1 | 3 | 1 | 2 | 1 | 3 |

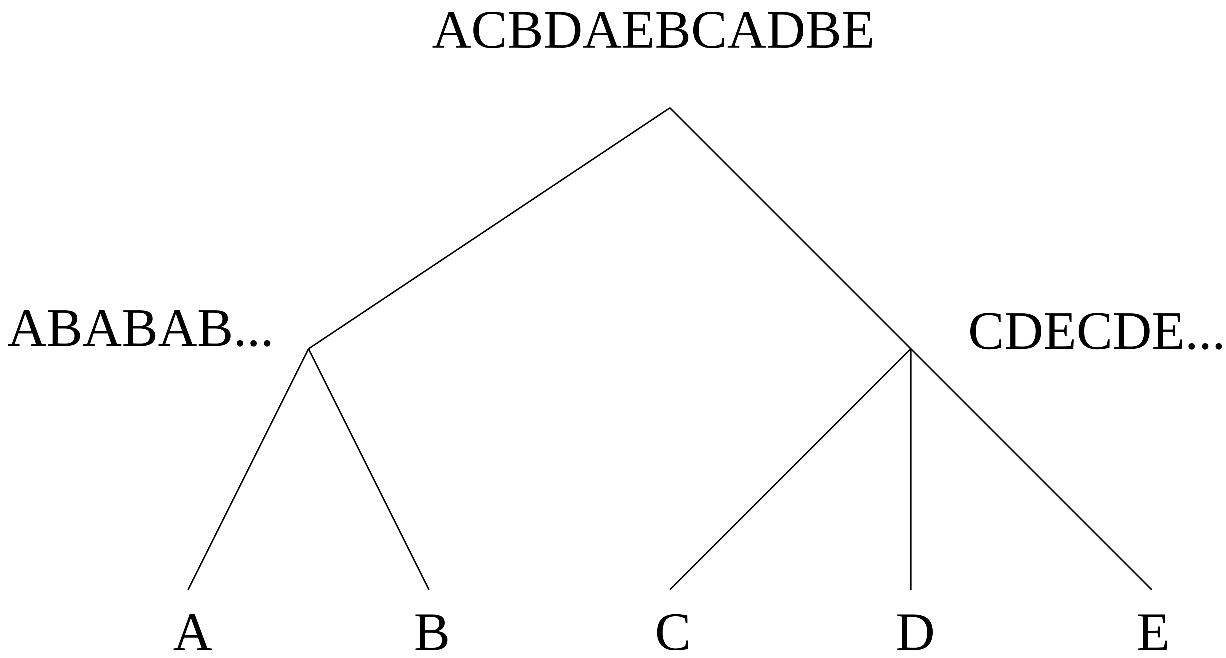
- A perfect solution with $h = 1$
  for either $W = \{2, 4, 4\}$ or $W = \{2, 4, 5\}$.

| | | | | | |
|---|---|---|---|---|---|
| 1 | 2 | 1 | 2 | 1 | 2 |
| 3 | 4 | 5 | 3 | 4 | 5 |

- A perfect solution for $W = \{2, 2, 3, 3, 3\}$ with $h = 2$.

- There is no solution for $W = \{2, 3, 6\}$ with $h = 1$.
  - Even though $1/2 + 1/3 + 1/6 = 1$.
  - No solution for $W = \{2, 3, 6\}$ for any $w \geq 1$.

# Tree to Schedule – Example




Observation:

The window size of a leaf label is the product of the degrees of all of its ancestors.

# A Tree Representation for Schedules

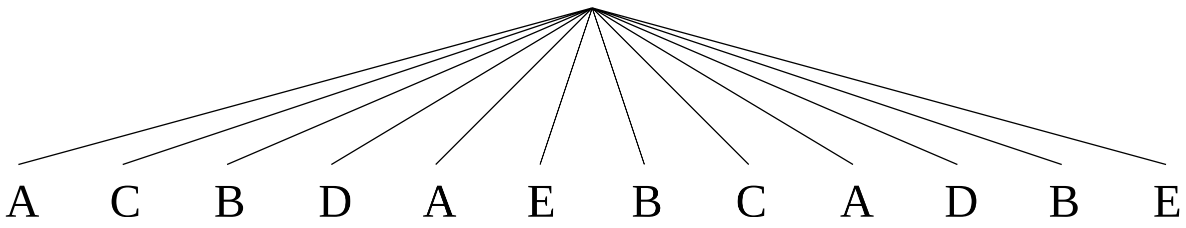
Tree to schedule:

- ⋆ Each leaf is associated with a label.
- ⋆ $T$: a tree whose sub-trees are $T_1, \ldots, T_d$.
- ⋆ $S_1, \ldots, S_d$: the schedules generated by the sub-trees.
- ⋆ W.o.l.g., the lengths of all the schedules are the same (using replication).
- ⋆ $S$ the schedule for $T$:
  - select in a round-robin fashion the labels from $S_1, \ldots, S_d$.

## Schedules to Trees

A schedule of length $n$ can be represented by a star tree where the root has $n$ leaf children.

Example:




Goal: condensed trees with $<< n$ leaves.

Remark: easier for perfect schedules.

## A Perfect Schedule With Only a Star Tree Representation

- Windows: $(6, 10, 15, 30, \ldots, 30)$.

- If the degree of the root is $2, 3, 5$ then one of the windows $15, 10, 6$ is impossible.

- A perfect schedule:

$$ABC---A----BA----CA--B--A------$$

# A Binary Tree Schedule

Property: In binary trees all the windows are of the form: $2^v$ for $v \geq 0$.

Lemma If $w_i = 2^{v_i}$ and $\sum \frac{1}{2^{v_i}} = 1$ then there exists a binary tree representing an optimal perfect schedule.

Proof:

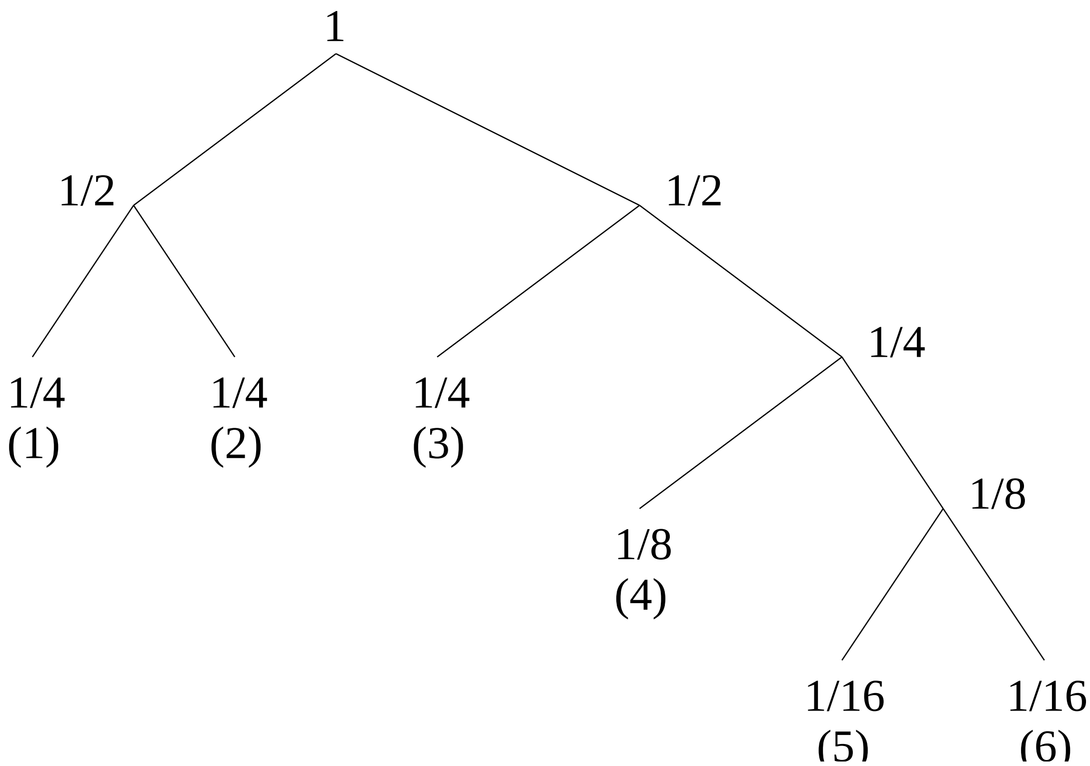
* Trivial for $w_1 = 1$.
* $2 \leq w_1 \cdots \leq w_n \Rightarrow \frac{1}{2} \geq \frac{1}{2^{v_1}} \geq \cdots \geq \frac{1}{2^{v_n}}$.
* There exists an index $1 \leq j < n$:

$$\sum_{i=1}^{j} \frac{1}{2^{v_i}} = \sum_{i=j+1}^{n} \frac{1}{2^{v_i}} = \frac{1}{2}$$

* Continue recursively with:

$\{w_1, \ldots, w_j\}$ and $\{w_{j+1}, \ldots, w_n\}$

# Example




Windows: $W = \{4, 4, 4, 8, 16, 16\}$.

Schedule:

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 3 | 2 | 4 | 1 | 3 | 2 | 5 | 1 | 3 | 2 | 4 | 1 | 3 | 2 | 6 |

# Binary Trees Schedules

Lemma If $w_i = 2^{v_i}$ and $\sum \frac{1}{2^{v_i}} = h(W)$ then there exist binary trees representing perfect schedules with $H(W) = \lceil h(W) \rceil$ rows.

Proof:

* There exists an index $1 \leq j \leq n$:

$$\sum_{i=1}^{j} \frac{1}{2^{v_i}} = 1$$

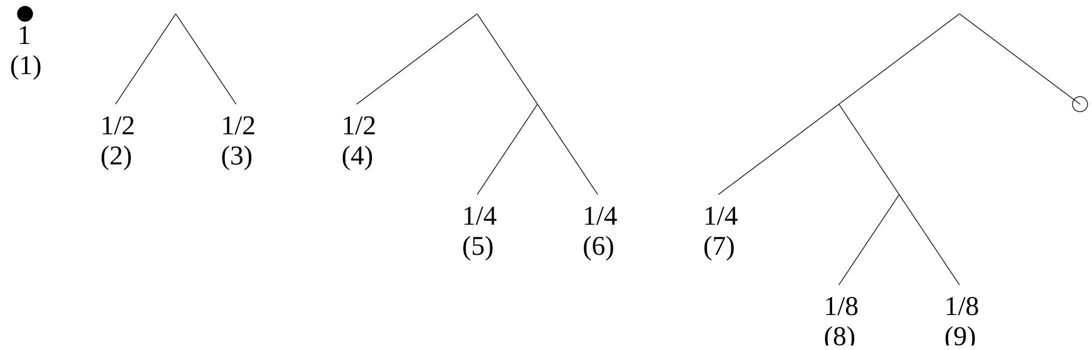
* Construct a binary tree for $\{w_1, \ldots, w_j\}$.
* Continue recursively with $\{w_{j+1}, \ldots, w_n\}$.
* There are $\lfloor h(W) \rfloor$ such full binary trees.
* $h(W)$ not an integer $\Rightarrow$ one non-full binary tree.

# Example

⋆ $W = \{1, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}\}$.

⋆ 3 full binary trees and 1 non-full binary tree:

$\{1\}$ $\{\frac{1}{2}, \frac{1}{2}\}$ $\{\frac{1}{2}, \frac{1}{4}, \frac{1}{4}\}$ $\{\frac{1}{4}, \frac{1}{8}, \frac{1}{8}\}$




⋆ Schedule:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | 3 | 2 | 3 | 2 | 3 | 2 | 3 |
| 4 | 5 | 4 | 6 | 4 | 5 | 4 | 6 |
| 7 | · | 8 | · | 7 | · | 9 | · |

## A $\delta$-Binary Tree Schedule

Definition: In a $\delta$-binary tree the root has $\delta$ binary sub-trees. ($\delta \geq 1$ is an integer.)

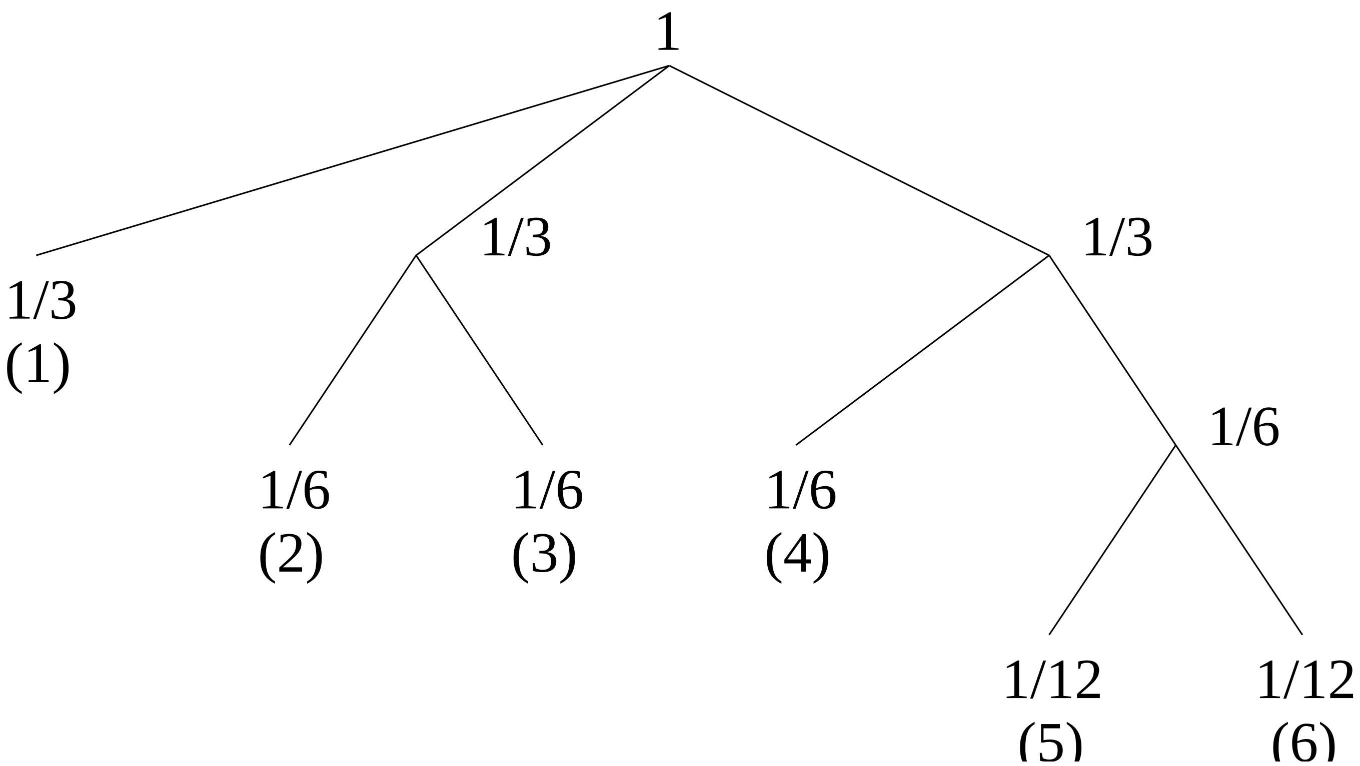
Property: In $\delta$-binary trees all the windows are of the form: $\delta 2^v$ for $v \geq 0$.

Lemma If $w_i = \delta 2^{v_i}$ and $\sum \frac{1}{w_i} = 1$ then there exists a $\delta$-binary tree representing an optimal perfect schedule.

Proof:

- $\delta \leq w_1 \cdots \leq w_n \Longrightarrow 1 \geq \frac{1}{\delta 2^{v_1}} \geq \cdots \geq \frac{1}{\delta 2^{v_n}}$.
- There exists an index $1 \leq j < n$: $\sum_{i=1}^{j} \frac{1}{\delta 2^{v_i}} = \frac{1}{\delta}$
- Construct a binary tree for $\{\frac{w_1}{\delta}, \ldots, \frac{w_j}{\delta}\}$.
- Continue recursively with: $\{w_{j+1}, \ldots, w_n\}$
- There are $\delta$ such binary trees.
- These $\delta$ trees are the children of the root.

## Example




Windows: $W = \{3, 6, 6, 6, 12, 12\}$.

Schedule:

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 1 | 3 | 5 | 1 | 2 | 4 | 1 | 3 | 6 |

# $\delta$-Binary Trees Schedules

Lemma If $w_i = \delta 2^{v_i}$ and $\sum \frac{1}{w_i} = h(W)$ then there exist $\delta$-binary trees representing perfect schedules with $H(W) = \lceil h(W) \rceil$ rows.

Proof:

* There exists an index $1 \leq j < n$:

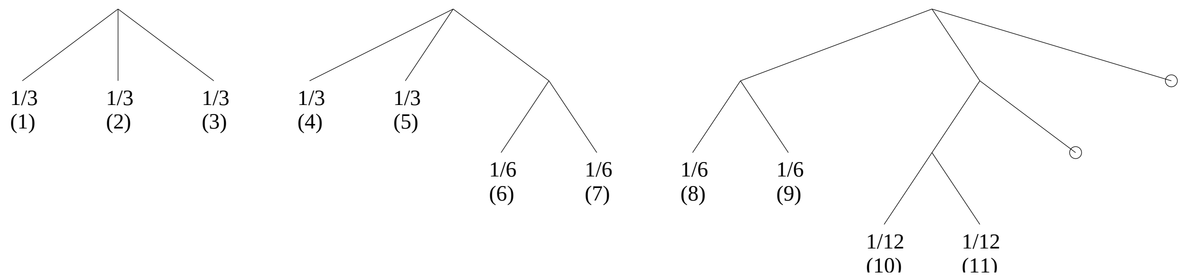
$$\sum_{i=1}^{j} \frac{1}{\delta 2^{v_i}} = 1$$

* Construct a $\delta$-binary tree for $\{w_1, \ldots, w_j\}$.
* Continue recursively with $\{w_{j+1}, \ldots, w_n\}$.
* There are $\lfloor h(W) \rfloor$ such full $\delta$-binary trees.
* $h(W)$ not an integer $\Rightarrow$ one non-full $\delta$-binary tree.

# Example

- $W = \left\{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{12}, \frac{1}{12}\right\}$.
- 2 full 3-binary trees and 1 non-full 3-binary tree:

$$\left\{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right\} \quad \left\{\frac{1}{3}, \frac{1}{3}, \frac{1}{6}, \frac{1}{6}\right\} \quad \left\{\frac{1}{6}, \frac{1}{6}, \frac{1}{12}, \frac{1}{12}\right\}$$




- Schedule:

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| 4 | 5 | 6 | 4 | 5 | 7 | 4 | 5 | 6 | 4 | 5 | 7 |
| 8 | 10 | · | 9 | · | · | 8 | 11 | · | 9 | · | · |

## An Asymptotic Solution

An additive approximation:

$$H(W) \leq h(W) + e\ln(h(W)) + 7.34$$

A multiplicative approximation:

$$H(W) = 1 + O\left(\frac{\ln(h(W))}{h(W)}\right).$$

# Main Ideas

Algorithm:

* Partition the windows into bins.
* Associate an integer value $\delta \geq 1$ to each bin.
* In the $\delta$-bin, round down the windows such that:
  - all the new windows are of the form $\delta 2^v$.
* Schedule optimally each bin.

Problem: How to find a good partition?

* Minimize the bandwidth loss by the rounding.
* Minimize the residual waste of the bins.

## Example

- $W = \{1, 2, 3, \ldots, 19\}$.
- 2-bin: $\{2, 4, 8, 8(9), 16, 16(17), 16(18), 16(19)\}$.
- 3-bin: $\{3, 6, 6(7), 12, 12(13), 12(14), 12(15)\}$.
- 5-bin: $\{5, 10, 10(11)\}$.
- One row for: $\{1\}$.
- One row for: $\{2, 4, 8, 8(9)\}$.
- One row for: $\{3, 6, 6(7), 12, 12(13), 12(14), 12(15)\}$.
- The remaining windows are $\{5, 10, 11, 16, 17, 18, 19\}$.
- 2-bin: $\{4(5), 8(10), 8(11), 16, 16(17), 16(18), 16(19)\}$.

# Example - Schedule

Windows: $W = \{1, 2, 3, \ldots, 19\}$.

Schedule:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | 4 | 2 | 8 | 2 | 4 | 2 | 9 |
| 3 | 6 | 12 | 3 | 7 | 13 | 3 | 6 |
| 5 | 10 | 16 | · | 5 | 11 | 17 | · |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | 4 | 2 | 8 | 2 | 4 | 2 | 9 |
| 14 | 3 | 7 | 15 | 3 | 6 | 12 | 3 |
| 5 | 10 | 18 | · | 5 | 11 | 19 | · |

## Asymptotic Solution – More Details

Recursive lemma:

* Fix an integer $k \geq 1$.
* There exists a perfect schedule whose number of rows is no more than

$$h(W) + kh(W)^{\frac{1}{k}} + 2k - 1.$$

* $k = 1 \Rightarrow$
  $$H(W) \leq 2h(W) + 1.$$
* $k = 2 \Rightarrow$
  $$H(W) \leq h(W) + 2\sqrt{h(W)} + 3.$$
* $k = \lceil \ln(h(W)) \rceil \Rightarrow$
  $$H(W) = h(W) + O(\ln(h(W))).$$

## Recursive Lemma for $k = 1$

- Let $2^{v_i} \leq w_i < 2^{v_i+1}$.
- Let $W' = \{2^{v_1}, \ldots, 2^{v_n}\}$ (A 2-bin).
- There exists a perfect schedule for $W'$ with $\lceil h(W') \rceil$ rows.
- $h(W') = \sum \frac{1}{2^{v_i}} < 2 \sum \frac{1}{w_i} = 2h(W)$.
- Thus, at most $\lceil 2h(W) \rceil \leq 2h(W) + 1$ rows are used in the schedule.
- We call this schedule $S_1$.

## Recursive Lemma for $k > 1$

- Partition $W$ into $x$ bins: $B_x, \ldots, B_{2x-1}$.
- Case $w_i < x$:
  - Let $x \leq 2^{v_i} w_i = u < 2x$.
  - Put $i$ in the bin $B_u$.
  - Let $w'_i = w_i$.
- Case $w_i \geq x$:
  - Let $2^{v_i} u \leq w_i < 2^{v_i}(u+1)$ for $x \leq u < 2x$.
  - Put $i$ in the bin $B_u$.
  - Let $w'_i = 2^{v_i} u$.
- $Z_u$ is a $u'$-bin for $u = u' 2^v$.
- Each bin has a perfect schedule for $w'_i$ except for a residual whose sum is less than $1$.
- $W_1$: windows scheduled perfectly according to $w'_i$.
- $W_2$: the rest of the windows.
- Use $S_{k-1}$ to schedule $W_2$.
- The proof follows by choosing: $x = \left\lfloor h(W)^{(k-1)/k} \right\rfloor$.

# Harmonic Windows Scheduling

Input: $w_i = i$ for $i = 1, 2, \ldots, n$.

A dual optimization goal:

⋆ $N(h)$: the maximum $n$ such that there exists a schedule for $W = \{1, 2, \ldots, n\}$ on $h$ rows.

Lower bound:

⋆ $H(n) \geq$ the harmonic number $H_n = \sum_{i=1}^{n} \frac{1}{n}$.

Upper bound:

⋆ $N(h) \leq \alpha e^h$ for $\alpha = e^{-0.57721} \approx 0.56147$.

Asymptotic bounds:

⋆ $N(h) \geq 0.00035 \frac{e^h}{h^e}$.

⋆ $N(h) \geq \left(e - O\left(\frac{\ln(h)}{h}\right)\right)^h$

– Better than $(e - O(1))^h$.

# Examples

$h = 1$ and $n = 1$:

| | | |
|---|---|---|
| 1 | 1 | $\ldots$ |

$h = 2$ and $n = 3$:

| | | | |
|---|---|---|---|
| 1 | 1 | 1 | 1 |
| 2 | 3 | 2 | 3 |

$h = 3$ and $n = 9$:

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | 4 | 2 | 5 | 2 | 4 | 2 | 5 | 2 | 4 | 2 | 5 |
| 3 | 6 | 8 | 3 | 7 | 9 | 3 | 6 | 8 | 3 | 7 | 9 |

Remark:

* Impossible to schedule $\{1, \ldots, 10\}$ on 3 rows.
* Although $\frac{1}{1} + \frac{1}{2} + \cdots + \frac{1}{10} \leq 3$.
* Even with a non-perfect schedule.
* Even if $i$ may appear in more than one row.

# The Greedy Algorithm

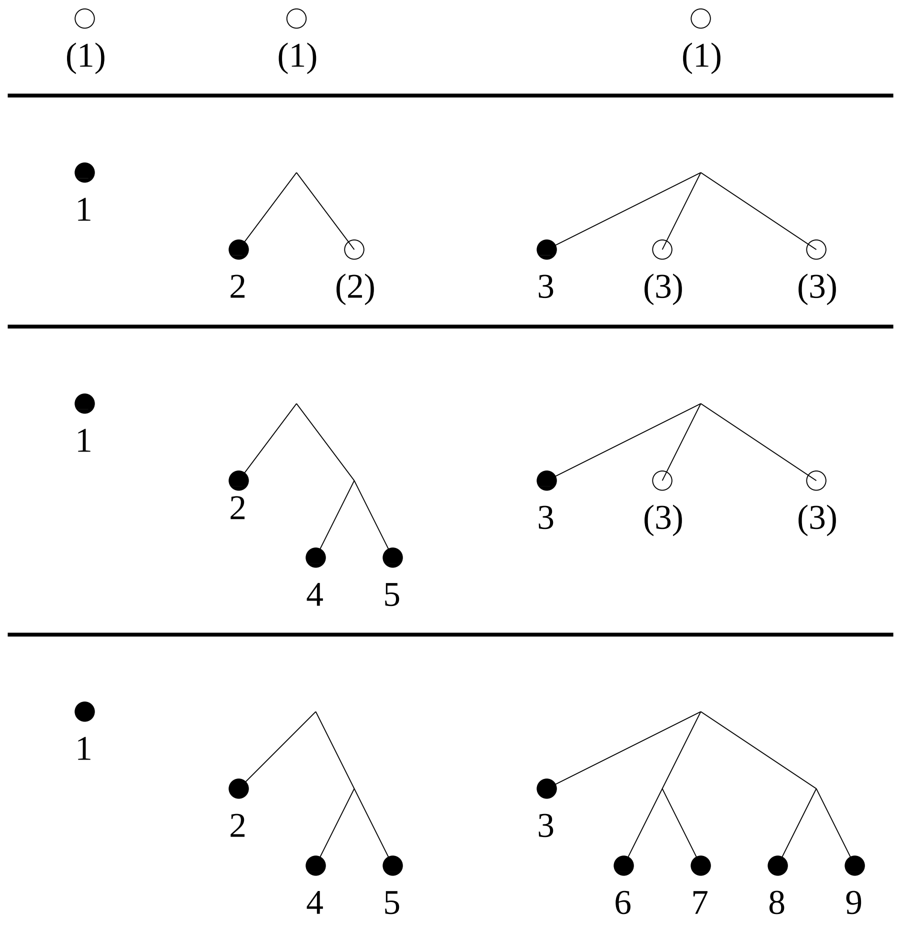
General scheme: In round $1 \leq r \leq n$, a perfect placement for $r$ is found.

The greedy rule:
- Minimize the difference $r - r'$ where $r'$ is the gap between placements of $r$.
- In each round minimize the bandwidth waste $\frac{1}{r'} - \frac{1}{r}$.

Data structure:
- ⋆ Each of the $h$ schedules is represented by a tree.
- ⋆ Some leaves are closed being assigned to numbers.
- ⋆ In round $r$:
  - Either $r$ is assigned to an existing open leave and thus closing it.
  - Or one of the leaves is split and $r$ is assigned to one of its descendent leaves, closing this leaf and leaving all the other descendent leaves open.

Termination rule: Terminate when $r$ cannot be assigned to any open leave.

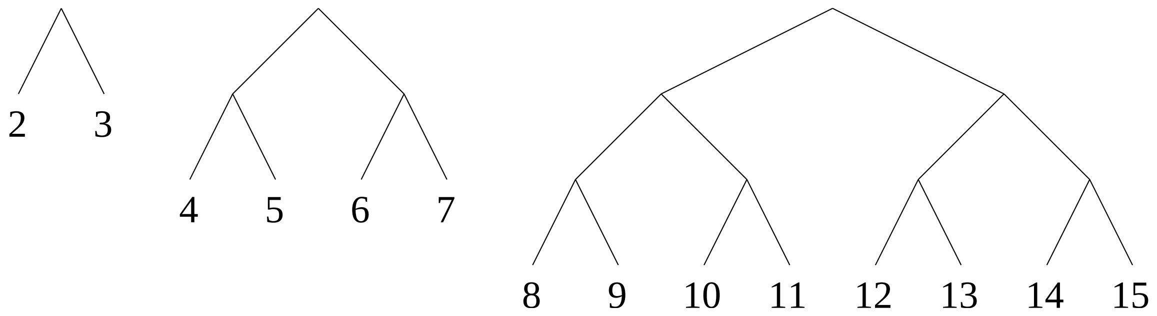
# The Greedy Algorithm for $h = 3$



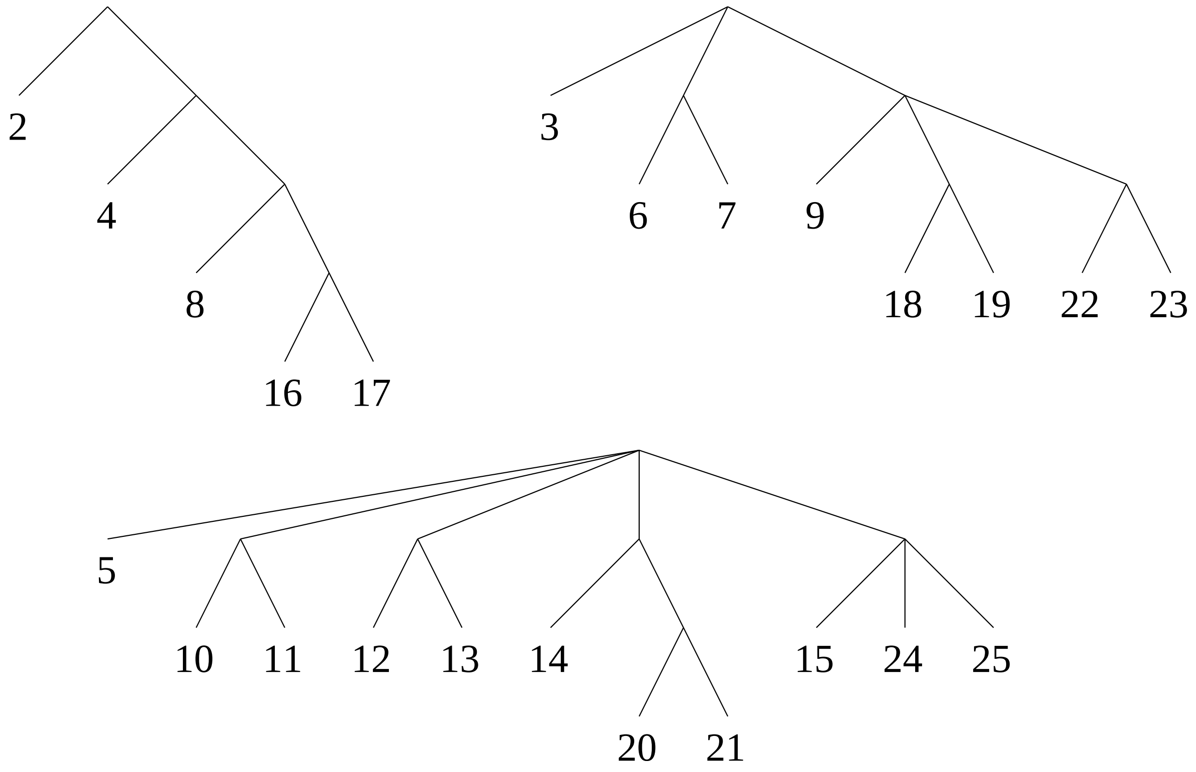
## The $h = 4$ Case

- Simple solution: $n = 15$
  - A special case of $n = 2^h - 1$.
- Greedy: $n = 25$.
- A modified greedy: $n = 26$.
  - The best perfect schedule.
- Best non-perfect schedule: $n = 28$.
- The upper bound: $n = 30$.

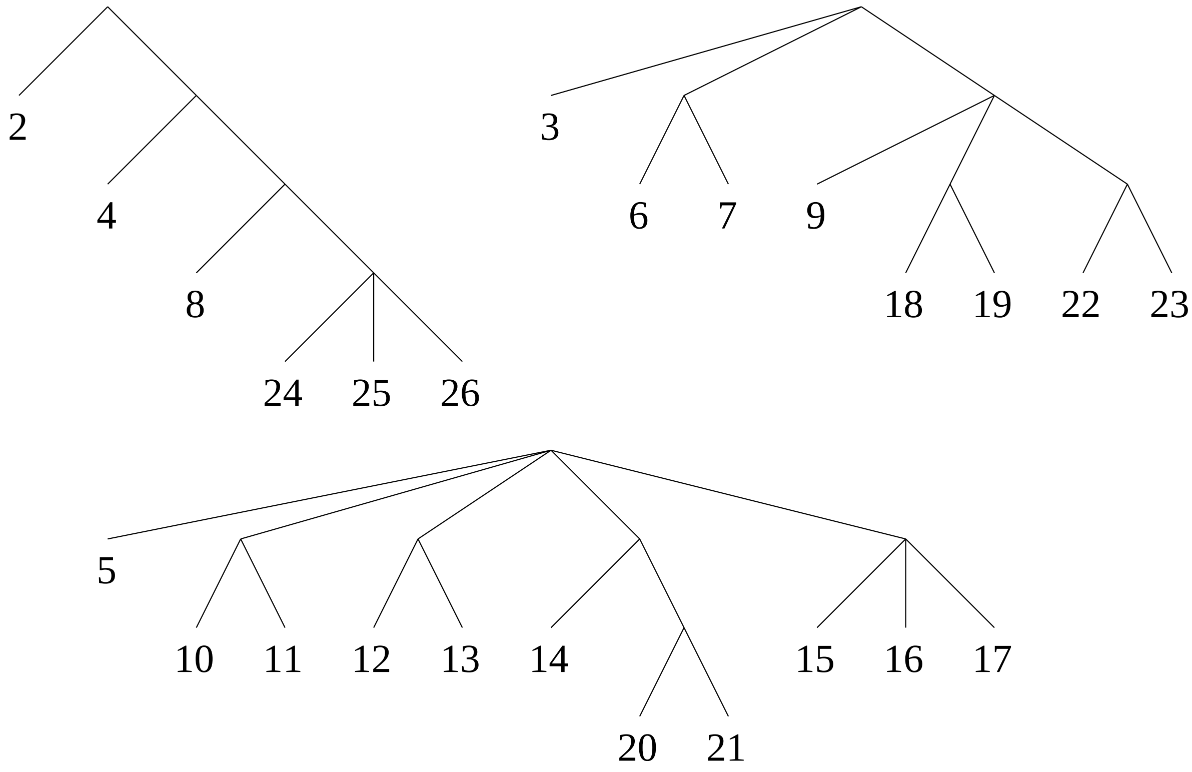
# The $n = 2^h - 1$ Solution for $h = 4$



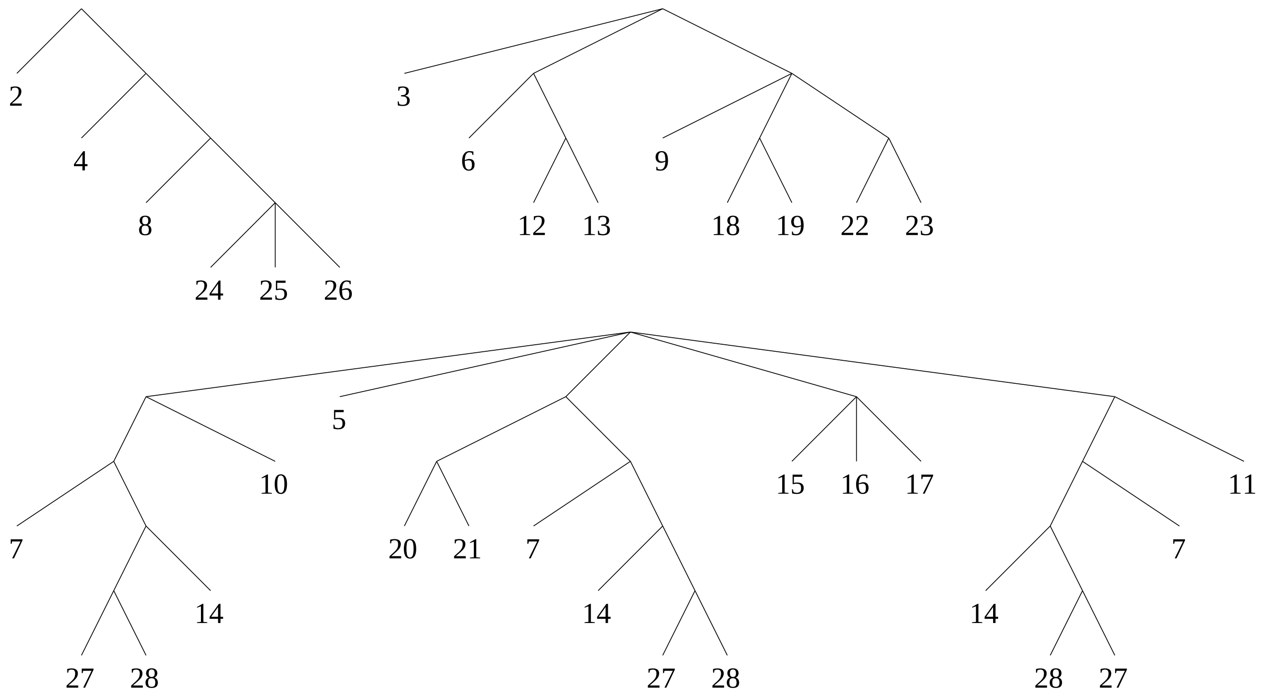
# Greedy for $h = 4$ and $n = 25$



## Modified Greedy for $h = 4$ and $n = 26$



# A Non-Perfect Schedule for $h = 4$ and $n = 28$



## Solutions for small $h$

| $h$ | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Upper bound | 1 | 3 | 10 | 30 | 82 | 226 |
| Best bound | 1 | 3 | 9 | 28 | 77 | 211 |
| Greedy | 1 | 3 | 9 | 26 | 73 | 204 |
| Best perfect | 1 | 3 | 9 | 26 | 75 | 207 |
| $2^h - 1$ | 1 | 3 | 7 | 15 | 31 | 63 |

| 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|
| 615 | 1673 | 4549 | 12366 | 33616 | 91379 |
| 570 | 1573 | 4325 | 11759 | 31677 | 86428 |
| 570 | 1573 | 4325 | 11759 | 31677 | 86428 |
| 570 | 1573 | 4325 | 11759 | 31677 | 86428 |
| 127 | 255 | 511 | 1023 | 2047 | 4095 |

Maximum Greedy: $N(20) = 265133522$.

– $2.74\%$ of the $272400599$ upper bound.

# A Combination Technique

A combination theorem:

Combining $\langle h_1, n_1 \rangle$ with $\langle h_2, n_2 \rangle$
$\Rightarrow \langle h_1 + h_2, (n_1 + 1)(n_2 + 1) - 1 \rangle$

An asymptotic corollary:

A repeat combination of $\langle k, n \rangle$ with itself
$\Rightarrow \left\langle h, \left(\sqrt[k]{n+1}\right)^h - 1 \right\rangle$-schedule.

Examples:

⋆ Combining $\langle 3, 9 \rangle$ with $\langle 4, 28 \rangle$
$\Rightarrow \langle 7, 289 \rangle$.

⋆ A repeat combination of $\langle 3, 9 \rangle$ with itself
$\Rightarrow \left\langle h, \sqrt{10}^h - 1 \right\rangle$.

Comparisons:

⋆ Greedy outperforms any combination.

⋆ The combination technique outperforms the asymptotic result up-to some constant.

## Windows Scheduling Open Problems

- ⋆ Improve the asymptotic lower bound for harmonic windows scheduling.

- ⋆ Improve the lower bounds for small $h$ for harmonic windows scheduling.

- ⋆ Prove bounds for the greedy algorithm:
  - for the harmonic windows scheduling.
  - for the general windows scheduling.

- ⋆ Learn the power of:
  - Imperfect schedules.
  - Numbers appear in more than one row.

## Guaranteed Delay for Air-Cashing

A server broadcasts on several channels equal size information-pages belonging to providers.

Clients listen to the channels until they find the page they wish to load.

The server sells the provider of page $i$ a guaranteed delay $w_i$ (Quality-of-Service).

If the number of slots between two consecutive appearances of page $i$ is no more than $w_i - 1$ then clients never wait more than $w_i$ slots to get their information.

The goal is to respect all the contracts with as minimum as possible channels.

Or, for a given number of channels, the server tries to sign profitable contracts.

# Media-on-Demand with Buffering

A server broadcasts a program over $h$ channels.

The program is partitioned into $n$ equal size segments.

Clients can receive data from all channels.

Clients buffer segment $i$ the first time they see it.

Clients start a receiving and buffering process at a beginning of a segment transmission.

The harmonic windows scheduling guarantees that the number of slots between two consecutive appearances of segment $i$ is no more than $i - 1$.

The start-up delay for playing the program with no-interruption is at most the length of the segment.

That is, $1/n$ the length of the transmission.

## Push if You Cannot Pull!

Pull systems:

⋆ Send request - get answer.

⋆ Immediate answers, not with many requests.

Push systems:

⋆ Periodically answers to popular requests.

⋆ Wait for the answer.

Tradeoff:

⋆ Delay vs. Scalability.

## Media-on-Demand

Media types:

- Full day broadcasting event.
- 2 hour movie.
- 10 minute song.
- 2 minute news clip.

Servers:

- Cable companies.
- Internet.
- Cellular phones companies.

Goals:

- Uninterrupted playback with small start-up delay.
- Minimizing the server bandwidth.
- Simple systems for the servers and the clients.

# A New Approach

Existing systems:

* Clients view the whole program from a single channel.
* Batching with multicasting imply a tradeoff between the start-up delay and the server bandwidth.

The Pyramid paradigm:

* Clients receive data simultaneously from more than one channel.
* Clients buffer segments of the program to view later.
* An exponentially better tradeoff between the start-up delay and the server bandwidth.

## Start-Up Delay

- ⋆ Without buffering and larger receiving capacity the best the server can do is to stager the transmission on the $h$ channels.
- ⋆ This guarantees a start-up delay of $1/h$ the program length.

Comparison with harmonic windows scheduling:

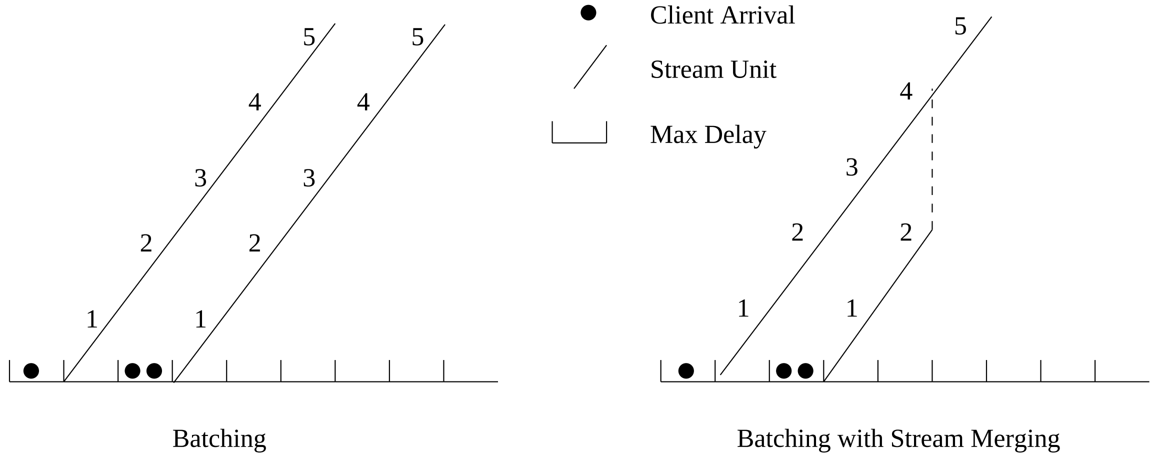
– For a two hours movie ($120$ minutes).

– For $1$ to $6$ broadcasting channels.

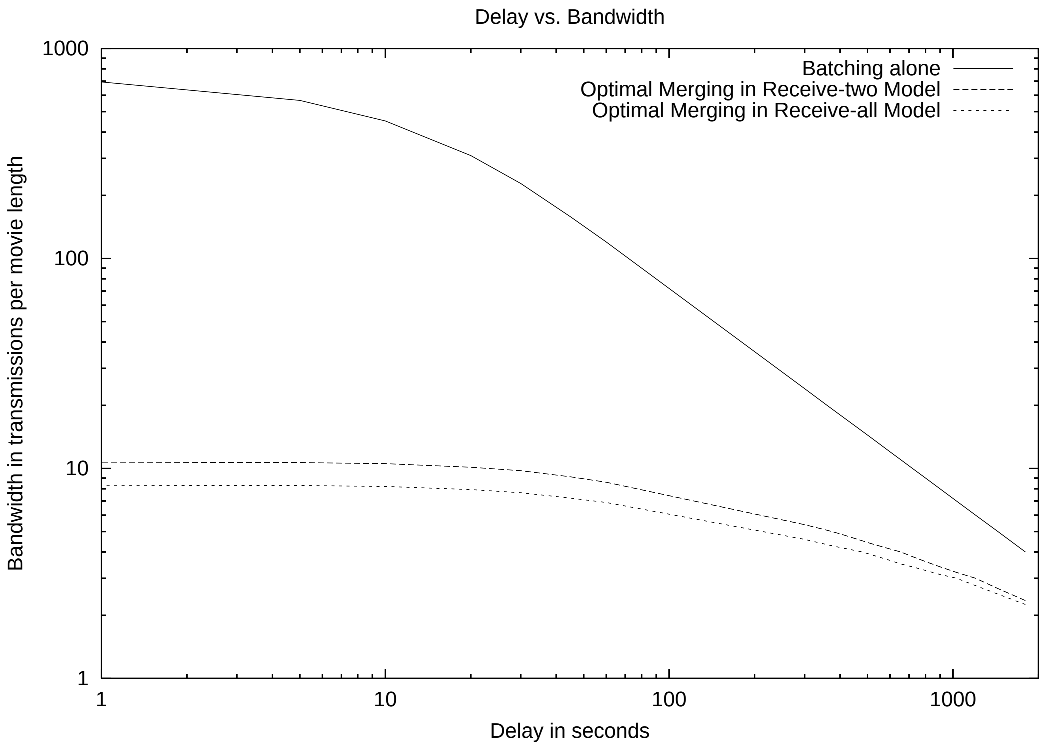
| # of channels | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| delay w.o. buffers | 120 | 60 | 40 | 30 | 24 | 20 |
| delay w. buffers | 120 | 40 | 13.33 | 4.29 | 1.56 | 0.57 |

# The Stream Merging Model

- A good technique for on-line scenarios.
- An adaptive solution for many programs that change their popularity.



## The Bandwidth Start-Up Delay Tradeoff




- Comparison of bandwidth required for batching and batching with optimal stream merging in which:
  - Clients can receive data from two channels.
  - Clients can receive data from all channels.

- The figure plots the bandwidth requirement vs. delay:
  - For a $2$ hour movie ($7200$ seconds).
  - With Poisson arrivals averaging every $10$ seconds.

## The Broadcast Disks Problem

The servers are the providers.

Servers choose the frequency in which each page is broadcast.

Applied in uni-directional environments:

⋆ Caching web pages in the air.

⋆ Teletext.

⋆ Wireless communication.

# The Basic Model

Input:

* $n$ equal size pages: $A_1, A_2, \ldots, A_n$.
* Access probabilities (popularities):
  - $p_1, p_2, \ldots, p_n$.
  - $\sum p_i = 1$.

Output:

* A cyclic slotted schedule:
  - $i_1, i_2, \ldots, i_T, i_1, i_2, \ldots$
  - $i_j \in \{1, \ldots, n\}$.

Model:

* Arrivals of clients are uniformly (Poisson) distributed.

Goal:

* Minimize the expected waiting time.

## Computing the Expected Waiting Time

- Clients arrive at the beginning of slots and depart at the end of the first slot their page is transmitted.

- For $1 \leq t \leq T$ and $1 \leq i \leq m$:

  – $S(i,t) :=$ number of time slots a client arriving at slot $t$ needs to wait until $A_i$ is transmitted.

- The average waiting time is:

$$W = \sum_{i=1}^{m} p_i \frac{\sum_{t=1}^{T} S(i,t)}{T}$$

## Example

2 pages A and B: $p_A = \frac{4}{5}; \quad p_B = \frac{1}{5}$

⋆ $A, A, A, A, B, A, A, A, A, B, \ldots$

$$W = \frac{4}{5} \cdot \frac{1+1+1+1+2}{5} + \frac{1}{5} \cdot \frac{1+2+3+4+5}{5}$$

$$W = \frac{39}{25} \approx 1.56 > 1.5$$

⋆ $A, B, A, B, \ldots$

$$W = \frac{1+2}{2} = 1.5$$

⋆ $A, A, B, A, A, B, \ldots$

$$W = \frac{4}{5} \cdot \frac{1+1+2}{3} + \frac{1}{5} \cdot \frac{1+2+3}{3}$$

$$W = \frac{22}{15} \approx 1.47 < 1.5$$

## Another Example

3 pages A and B: $p_A = \frac{9}{14}; \quad p_B = \frac{4}{14}; \quad p_C = \frac{1}{14}$

$\star$ $A, B, A, C, A, B, A, B, A, C, A, B, \ldots$

$$W = \frac{9}{14} \cdot \frac{3(1+2)}{6} + \frac{4}{14} \cdot \frac{2+1+4+3+2+1}{6} + \frac{1}{14} \cdot \frac{1+\cdots+6}{6}$$

$$W = \frac{11}{6} \approx 1.83$$

$\star$ $A, B, A, C, B, A, A, B, A, C, B, A, \ldots$

$$W = \frac{9}{14} \cdot \frac{1+2+1+3+2+1}{6} + \frac{4}{14} \cdot \frac{1+2+3}{3} + \frac{1}{14} \cdot \frac{1+\cdots+6}{6}$$

$$W = \frac{53}{28} \approx 1.89$$

$\star$ $A, B, C, A, B, C, \ldots$

$$W = \frac{1+2+3}{3} = 2$$

Round Robin for $m$ pages: $A_1, A_2, \ldots, A_m, A_1, \ldots$

$$W_{RR} = \frac{1+2+\cdots+m}{m} = \frac{m+1}{2}$$

# Perfect Spacing

- ⋆ Perfect spacing - The gap between any $2$ appearances of $A_i$ in the schedule is fixed.
- ⋆ Following the popularities: the gap is $\frac{1}{p_i}$.
- ⋆ Impossible for $p_1 = \frac{1}{2};\ p_2 = \frac{1}{3};\ \ldots$
- ⋆ $W = \sum p_i \cdot \frac{\frac{1/p_i(1/p_i+1)}{2}}{1/p_i}$

$$= \sum p_i \cdot \frac{1+p_i}{2p_i}$$

$$= \frac{m}{2} + \frac{1}{2} = W_{RR}$$ **!!**

## A Lower Bound

- Perfect spacing following $q_1, \ldots, q_m$:
  - $q_i = \frac{\sqrt{p_i}}{\sum_{j=1}^{m} \sqrt{p_j}}$
- A fractional bound
  since the gap $\frac{1}{q_i}$ may be a fraction.
- Moreover, it is sometimes impossible.
  For example, $1/2$ and $1/3$.
- Solutions approximate the fractional bound
  not the optimal bound.

# FD vs. TD

- ⋆ The goal is to schedule tasks as perfect as possible according to some frequency vector ($q_i$s).

- ⋆ This is the problem of implementing a Frequency Division by a Time Division.

  – The chairperson assignment problem.

- ⋆ Various possibilities for the quality:

  – Minimizing some function of the frequencies (square for the waiting time).

  – Being as close as possible to the frequencies in any prefix of the schedule.

  – Perfect spacing with frequencies as close as possible to the demand frequencies.

# Approximation Solutions

- ⋆ $2$-approximation by rounding down each $q_i$ to a power of half.

- ⋆ $2$-approximation of a random schedule that follows the weighted probabilities $q_i$.

- ⋆ $2$-approximation of a greedy strategy that schedules the page that was starved the most weighted by its importance $q_i$.

- ⋆ $2$-approximation of a greedy strategy that tries to follow the shares $q_i$ for any prefix of the schedule.

- ⋆ $(9/8)$-approximation for a schedule that is based on the golden-ratio sequence. The gap can have at most three values around the desired value of $1/q_i$.

- ⋆ $(1 + \varepsilon)$-approximation polynomial time scheme.

## The Powers of Half Approximation

- ⋆ $r_i = \frac{1}{2^{k_i}}$ s.t. $\frac{1}{2^{k_i}} \leq q_i < \frac{1}{2^{k_i - 1}}$.
- ⋆ Hence, $q_i < 2r_i$.
- ⋆ For $r_1, \ldots, r_m$ there exists a perfect schedule in which the gap between any $2$ appearances of page $A_i$ is $2^{k_i} = \frac{1}{r_i}$.

Analysis:

$$
\begin{aligned}
W &= \sum p_i \cdot \frac{\frac{1/r_i(1/r_i+1)}{2}}{1/r_i} \\
&= \sum p_i \cdot \frac{1+r_i}{2r_i} \\
&= \frac{1}{2} + \sum \frac{p_i}{2r_i} \\
&< \frac{1}{2} + \sum \frac{p_i}{q_i} \\
&< 2 \cdot LB.
\end{aligned}
$$

## The Randomized Schedule

Scheduling rule:

– Choose the next page according to the weighted probabilities $q_i$.

Analysis:

$$\begin{aligned} W &= \sum p_i \left(1 \cdot q_i + 2(1-q_i)q_i + \cdots\right) \\ &= \sum p_i q_i \sum_{j=0}^{\infty} (j+1)(1-q_i)^j \\ &=^{\star} \sum p_i q_i \cdot \frac{1}{q_i^2} \\ &= \sum \frac{p_i}{q_i} \\ &= 2 \cdot LB - 1. \end{aligned}$$

$\star$ $\sum_{j=0}^{\infty} (j+1)x^j = \frac{1}{(1-x)^2}$

# Greedy I – Dynamic

⋆ $w_i(t)$ – #slots $A_i$ was not scheduled before slot $t$.

⋆ Schedule $A_i$ at $t$ for maximal $r_i = (1 + w_i(t))q_i$.

Example – $P = \left(\frac{9}{14}, \frac{5}{14}, \frac{1}{14}\right) \Rightarrow Q = \left(\frac{1}{2}, \frac{1}{3}, \frac{1}{6}\right)$

1. $r = \left(\frac{1}{2}, \frac{1}{3}, \frac{1}{6}\right) \Rightarrow S = (A)$
2. $r = \left(\frac{1}{2}, \frac{2}{3}, \frac{1}{3}\right) \Rightarrow S = (A, B)$
3. $r = \left(1, \frac{1}{3}, \frac{1}{2}\right) \Rightarrow S = (A, B, A)$
4. $r = \left(\frac{1}{2}, \frac{2}{3}, \frac{2}{3}\right) \Rightarrow S = (A, B, A, B)$
5. $r = \left(1, \frac{1}{3}, \frac{5}{6}\right) \Rightarrow S = (A, B, A, B, A)$
6. $r = \left(\frac{1}{2}, \frac{2}{3}, 1\right) \Rightarrow S = (A, B, A, B, A, C)$
7. $r = \left(1, 1, \frac{1}{6}\right) \Rightarrow S = (A, B, A, B, A, C, A)$
8. $r = \left(\frac{1}{2}, \frac{4}{3}, \frac{1}{3}\right) \Rightarrow S = (A, B, A, B, A, C, A, B)$
9. $r = \left(1, \frac{1}{3}, \frac{1}{2}\right)$

   $\Rightarrow T = 6$ and $S = (A, B, A, C, A, B)$

# Greedy II – Fair

- $f_i(t)$ – #of times $i$ appears in $i_1, i_2, \ldots, i_{t-1}$.
- $g_i(t) = t \cdot q_i - f_i(t)$.
- Schedule $A_i$ at $t$ for maximal $g_j(t)$.

Example – $P = \left(\frac{9}{14}, \frac{5}{14}, \frac{1}{14}\right) \Rightarrow Q = \left(\frac{1}{2}, \frac{1}{3}, \frac{1}{6}\right)$

1. $g = \left(\frac{1}{2}, \frac{1}{3}, \frac{1}{6}\right) \Rightarrow S = (A)$
2. $g = \left(0, \frac{2}{3}, \frac{1}{3}\right) \Rightarrow S = (A, B)$
3. $g = \left(\frac{1}{2}, 0, \frac{1}{2}\right) \Rightarrow S = (A, B, A)$
4. $g = \left(0, \frac{1}{3}, \frac{2}{3}\right) \Rightarrow S = (A, B, A, C)$
5. $g = \left(\frac{1}{2}, \frac{2}{3}, -\frac{1}{6}\right) \Rightarrow S = (A, B, A, C, B)$
6. $g = (1, 0, 0) \Rightarrow S = (A, B, A, C, B, A)$
7. $g = \left(\frac{1}{2}, \frac{1}{3}, \frac{1}{6}\right)$

   $\Rightarrow T = 6$ and $S = (A, B, A, C, B, A)$

## The Greedy Approach

⋆ $W \leq 2 \cdot LB$: a theoretical bound.

⋆ $W \rightarrow LB$: experimental results (usually the best).

⋆ Is Greedy a de-randomization of Random?

⋆ Is Greedy $< 2$ approximation to Opt?

## Simulations – the Zipf Distribution

* Approximating "real" world phenomena.
* $p_i = \frac{(1/i)^\theta}{\sum (1/j)^\theta}$
* $\theta = 0$ implies $p_1 = \cdots = p_m = \frac{1}{m}$.
* A skewed distribution as $\theta$ increases.
* For Internet access: $\theta = 0.8$

## The Golden Ratio Schedule

- Frequencies $f_1, \ldots, f_m$ such that $\sum f_i = 1$.
- $F_k$ a "large enough" Fibonacci number.
- $f_i \approx \frac{N_i}{F_k} = \frac{F_{k-j_i}+S_i}{F_k}$

**Theorem:** There exists a schedule, $T = F_k$:

* $S_i$ gaps of value $F_{j_i}$,
* $F_{k-j_i-2} + S_i$ gaps of value $F_{j_i+1}$,
* $F_{k-j_i-1} - S_i$ gaps of value $F_{j_i+2}$.

Indeed:

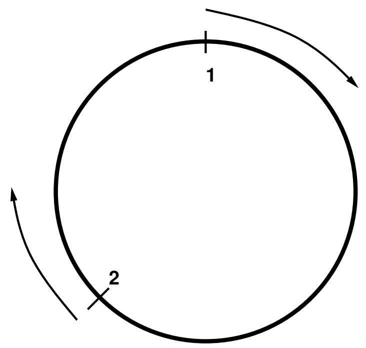
$$N_i = S_i + (F_{k-j_i-2} + S_i) + (F_{k-j_i-1} - S_i)$$

$$F_k = F_{j_i}S_i + F_{j_i+1}(F_{k-j_i-2} + S_i) + F_{j_i+2}(F_{k-j_i-1} - S_i)$$

$$F_{j_i} \le \frac{1}{f_i} \approx \frac{F_k}{N_i} \le F_{j_i+2}$$

# Constructing the Golden Ratio Schedule

* $\phi$ the golden ratio number ($\phi^2 = 1 + \phi$).
* $\phi \approx 1.618; \quad 1 - \phi = \frac{1}{\phi} \approx 0.618$
* $F_k = \frac{\phi^k + (1-\phi)^k}{\sqrt{5}}$



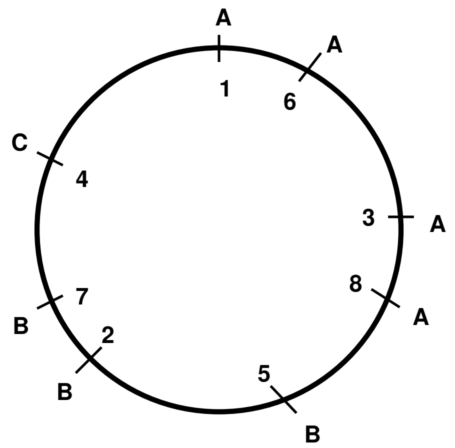

Construction:

(a) Start at point $1$.
(b) Walk clock-wise distance $\phi^{-1}$ to mark point $2$.
(c) Repeat $F_k - 1$ times (b) to mark points $1, \ldots, F_k$.
(d) Create a permutation of $1, \ldots, F_k$ starting at point $1$ going clock-wise back to point $1$.
(e) Page $A_1$ is scheduled in slots: $1, \ldots, N_1$.
(f) Page $A_i$ is scheduled in slots: $1 + \sum_{j=1}^{i-1} N_j, \ldots, \sum_{j=1}^{i} N_j$.

# Construction – Example

$P = (\frac{16}{26}, \frac{9}{26}, \frac{1}{26}) \Rightarrow Q = (\frac{1}{2}, \frac{3}{8}, \frac{1}{8})$

$F_k = 8 \qquad (f_A, f_B, f_C) = (\frac{4}{8}, \frac{3}{8}, \frac{1}{8})$




The permutation: $1, 6, 3, 8, 5, 2, 7, 4$

A's slots: $1, 3, 6, 8$

B's slots: $2, 5, 7$

C's slots: $4$

The GR schedule: $A, B, A, C, B, A, B, A \Rightarrow W = \frac{379}{208}$

A better schedule: $A, B, A, B, A, B, A, C \Rightarrow W = \frac{372}{208}$

The lower bound: $W \geq \frac{360}{208}$

## Analysis of the GR Schedule

Theorem

* GR is a $\frac{9}{8}$-approximation schedule.

Proof outline:

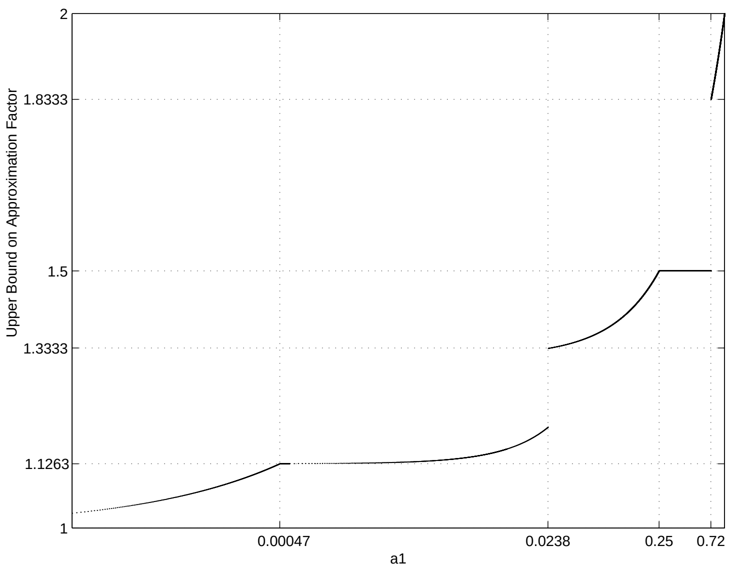
* LB $= \sum \left(\frac{1}{2q_i} + \frac{1}{2}\right) p_i$
* $W \leq \sum \left(\beta_i + \frac{1}{2}\right) p_i$
* $\frac{\beta_i + \frac{1}{2}}{\frac{1}{2q_i} + \frac{1}{2}} \leq \frac{9}{8}$ (Fibonacci manipulations).

# Approximations with Perfect Schedules

Let $q = \max\{q_i\}$, let $\rho$ be the approximation factor:

* $\rho \leq 1 + \sqrt[3]{24q}$ for all $q$.
* $\rho \leq \frac{9}{8-20q}$ for $q < \frac{1}{42} \approx 0.024$.
* $\rho \leq \frac{4}{3} + \frac{2}{3}q$ for all $q$.
* $\rho \leq \frac{3}{2}$ for $q < \frac{19-\sqrt{37}}{18} \approx 0.72$.

Each of these bounds is the best known for some $q$:



# What is a “Good” Perfect Schedule?

Parameters:

⋆ $f_i$ – the requested perfect frequency.

⋆ $g_i$ – the assigned perfect frequency.

⋆ $\rho_i = f_i/g_i$ – the loss or gain factor.

⋆ $\sigma_i = \max\{f_i/g_i, 1\}$ – the only loss factor.

Metrics:

⋆ Max: minimize $\max\{\rho_i\}$.

⋆ Weighted Max: minimize $\max\{f_i\rho_i\}$ or $\max\{f_i\sigma_i\}$.

⋆ Ave: minimize $\frac{1}{m}\sum \rho_i$ or $\frac{1}{m}\sum \sigma_i$.

⋆ Weighted Ave: minimize $\sum f_i\rho_i$.
(The broadcast disks problem.)

Remark: Weighted Max and Ave are not natural since $\rho_i = 1$ is not always the best.

## The Maintenance Service Problem

- ⋆ $m$ machines.
- ⋆ $M$ machines can be serviced at a time.
- ⋆ Operating cost – $(j+b)a_i$ for machine $i$ that was not scheduled $j$ slots.
- ⋆ Maintenance cost – $c_i$ for each slot machine $i$ is scheduled.

Goal: Find a schedule that minimizes the long-run average cost per time-slot.

The Broadcast Disks problem:

$$M = 1,\ b = 1,\ a_i = p_i,\ c_i = 0.$$

## Broadcast Disks Generalizations

* Each file is of length $l_i$ pages.
  - Files can or cannot be broadcast in parts.
  - Must or not receive a file from its beginning.
* There are $M$ disks.
  - Clients access all disks simultaneously.
  - Clients access all disks but not simultaneously.
  - Clients access only a subset of the disks.
* Each page has a broadcasting cost.
* Each page has a priority.
* Access dependencies among the pages.
* A non-linear objective function.

## Dependencies

* And:
  - For $X = \{i_i, i_2, \ldots, i_h\} \subseteq \{1, \ldots, m\}$
  - $p_X$ - prob. for accessing all $A_{i_1}, \ldots, A_{i_h}$.
* Or:
  - For $X = \{i_i, i_2, \ldots, i_h\} \subseteq \{1, \ldots, m\}$
  - $p_X$ - prob. for accessing one of $A_{i_1}, \ldots, A_{i_h}$.
* Order (Imply):
  - For $L = (i_1 < i_2 < \cdots < i_h)$
  - $p_L$ - prob. for accessing $A_{i_1}, ..., A_{i_h}$ in this order.
* Formula:
  - For $R$ a relation (e.g., $(i_1 \vee i_2) \wedge (i_3 \rightarrow i_1)$)
  - $p_R$ - prob. for accessing the pages following $R$.

## Broadcast Disks – 2 Pages

- Only: $p_A$, $p_B$, $p_{AB}$, $p_{A\to B}$, $p_{B\to A}$.
- Assume: $p_A \geq p_B$.
- $A$ appears in the cycle no less than $B$.
- $x_1, x_2, \ldots$ #appearances of $A$ between $B$s.

$$\Rightarrow \quad \{x_i\}_{i=1}^{\infty} \subseteq \{x, x+1\}$$

- $A^x B A^{x+1} B \geq \min\left\{A^x B \; ; \; A^{x+1} B\right\}$

Optimality: The optimal schedule is $A^x B$ for $x$ that is a function of all the $5$ probabilities.

- Also, for different length pages.
- Also, for different priorities.

## A Non-Linear Objective Function

Impatient clients:

* The cost of waiting $i$ slots is $i^k$ for $k > 1$ or even $e^i$.

Patient clients:

* The cost of waiting $i$ slots is $i^k$ for $k < 1$ or even $\ln(i)$.

Solutions:

* Generalizations of the linear case in which the cost of waiting $i$ slots is $i$.

A related problem:

* "Maximizing" the waiting time.

## The Clients-Providers-Servers Model

Environment:

* Servers have no freedom to select the data pages.
* Servers sell providers their broadcast capabilities.
  - Selling slots.
  - Selling shares (frequencies).
  - Selling guaranteed maximum access time.

Servers objective:

* Minimizing one of the access time objective functions.
* Respecting all the contracts with minimum resources (broadcast channels).
* Respecting most of the contracts.

# The Best Permutation Problem

Input:

- $n$ providers each gets one slot in a round-robin schedule ($W = \{n, n, \ldots, n\}$).
- There are pair-dependencies among the pages.

Goal: find the best permutation.

Results:

- For the Or And and Imply dependencies: the random permutation helps.
- This implies a randomized algorithm that can be de-randomized for certain values of $n$.

Related problem: The linear arrangement problem.

## Example

Problems with the $4$ pages $A, B, C, D$:

| | |
|---|---|
| AND | Most clients access $A$ AND $B$. |
| OR | Most clients access $A$ OR $B$. |
| IMPLY | Most clients access $A$ THEN $B$. |

Few clients have other access patterns:

- Only $A$, only $B$, ...
- $C$ AND $D$, $A$ OR $D$, ...

Schedules: $ABCD$, $BACD$, or $ACBD$.

Optimization:

- Arrivals in the beginning of slots ($4$ possibilities).
- Minimizing sum of the $4$ possible waiting times.

## Example

* AND:
  - $ABCD$: $2+4+4+3=13$.
  - $ACBD$: $3+4+3+4=14$.
* OR:
  - $ABCD$: $1+1+3+2=7$.
  - $ACBD$: $1+2+1+2=6$.
* IMPLY:
  - $ABCD$: $2+5+4+3=14$.
  - $BACD$: $5+4+7+6=22$.
  - $ACBD$: $3+6+5+4=18$.

## The Linear Arrangement Problem

- ⋆ Input: a graph $G = (V, E)$ and a weighted function $w : E \to \Re$.

- ⋆ Output: a linear ordering of the vertices of the graph
  – the linear arrangement.

- ⋆ New distance: $d(u, v)$ the linear distance between $u$ and $v$.

- ⋆ Objective: minimize $\sum w(u, v)d(u, v)$
  – sum of the weighted distances of all the edges.

- ⋆ The linear arrangement problem is NP-hard.

- ⋆ Best approximation ratio: $O(\log n)$.

- ⋆ A research idea: An approximation algorithm with an additive error of $o(n)$ would yield an algorithm with $1 + o(1)$ multiplicative approximation for the clients-providers-servers three problems.

**Push Systems Additional Problems**

* Tuning time vs. access time.
* Internal caching.
* The stretch measurement.
* Combining Pull and Push systems.
* Minimizing obsolete time of pages.