

Look at that sequence... Is it a vector? Is it a list?

No! It's a Super Tree!!

Martín Knoblauch Revuelta

<http://www.mkrevuelta.com>

@mkrevuelta

mkrevuelta@gmail.com

indizen  using std::cpp

Except where otherwise noted, this work is licensed under:  
<http://creativecommons.org/licenses/by-nc-sa/4.0/>



Presentation available in my semiabandoned blog:  
<http://www.mkrevuelta.com>

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Index

1. The problem
2. Super Tree
3. Non proportional view
4. Applications
5. Similar proposals
6. Let's think about it

Super Tree

Martín K.R.  
**indizen**

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Introduction to the problem

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Are lists evil?—Bjarne Stroustrup



<https://isocpp.org/blog/2014/06/stroustrup-lists>

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

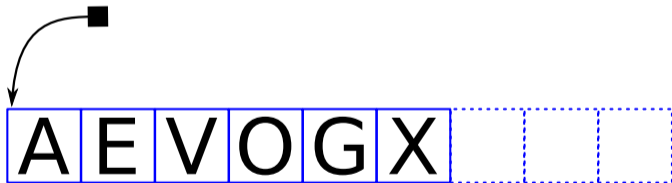
Applications

Similar  
proposals

Let's think  
about it

# Array

- Random access is fast
- Insertion/extraction are... slow



Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

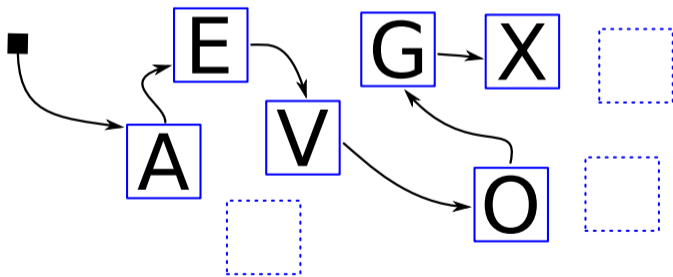
Applications

Similar  
proposals

Let's think  
about it

# Linked list

- Insertion/extraction are fast
- Random access is... sloooooow



Super Tree

Martín K.R.  
indizen

Intro

Super Tree

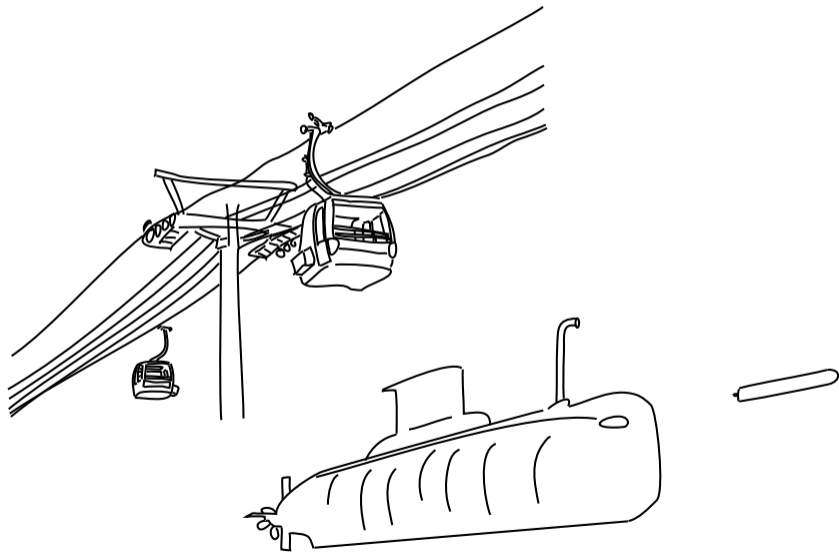
Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# How to compare them?



Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it



# Jon Bentley's suggestion



```
for (;;)
{
```

Random  
access

Insertion /  
extraction

```
}
```

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

## Jon Bentley's suggestion

“Insert a sequence of random integers  
into a sorted sequence,

then **remove** those elements **one by one**  
as determined by  
a **random** sequece of **positions**”

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

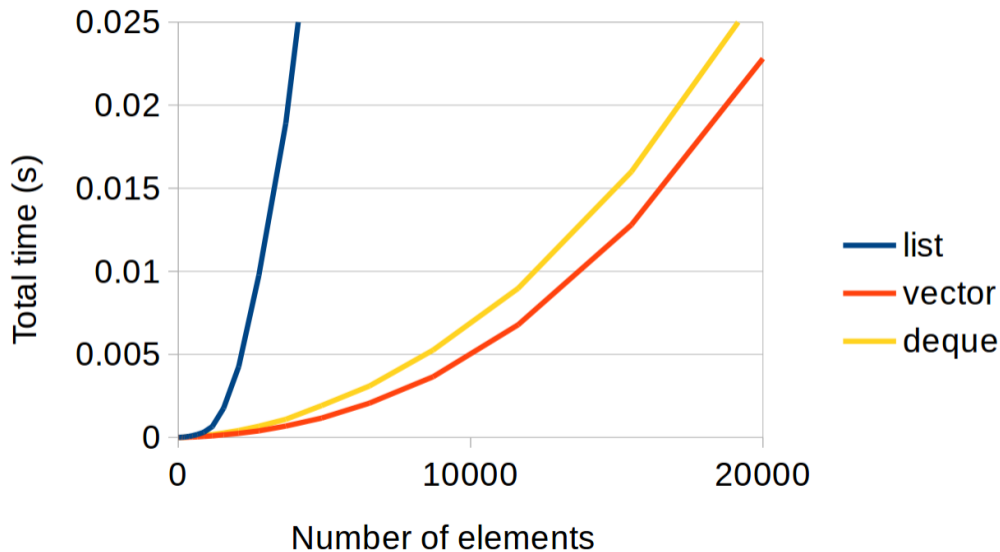
Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Results



Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Conclusion

Vectors are faster  
by some fixed proportion  
(a considerable proportion)

**But...**

Are we really interested  
in Jon Bentley's problem?

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Super Tree

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

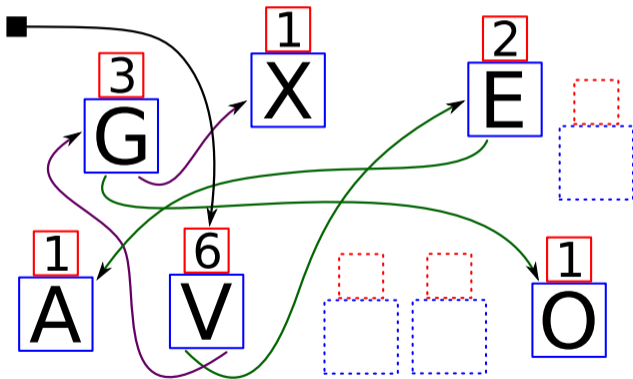
Applications

Similar  
proposals

Let's think  
about it

# Augmented tree (messed up)

Like a list, but with two "next"s (left, and right)



Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

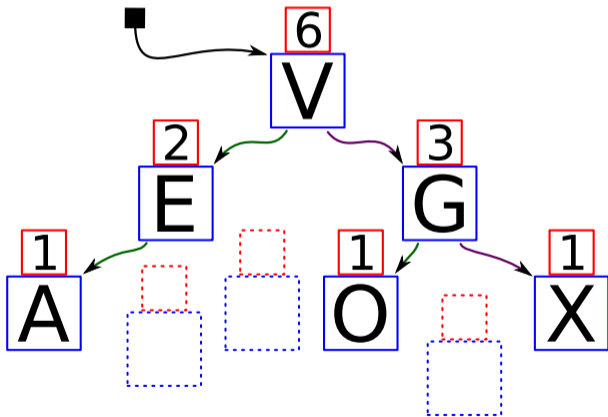
Applications

Similar  
proposals

Let's think  
about it

# Augmented tree

Special metadata: number of nodes in the sub-tree



Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Random access (1/3)

```
template <typename T>
struct node
{
    node<T> * left;           // Left sub-tree
    node<T> * right;        // Right "   "
    std::size_t count;      // Num. of nodes
    T value;                // Payload
};
```

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it



## Random access (2/3)

```
template <typename T>
node<T> * RandomAccess (node<T> * root,
                        std::size_t pos)
{
    if (pos >= root->count)
        return nullptr;

    node<T> * p = root;
```

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Random access (3/3)

```
for (;;)
{
    std::size_t nLeft = p->left ?
                        p->left->count : 0;

    if (pos == nLeft)           return p;
    else if (pos < nLeft)      p = p->left;
    else // (pos > nLeft)
    {
        pos -= nLeft + 1;
        p = p->right;
    }
} } // end
```

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

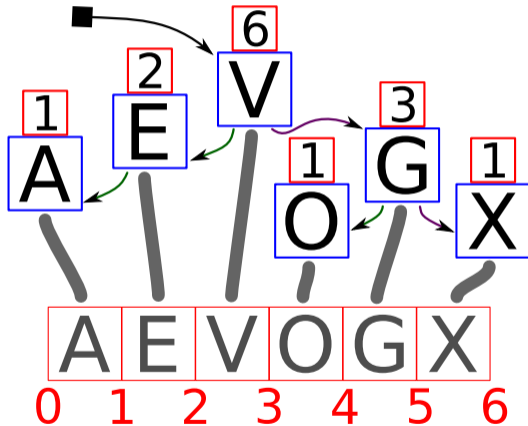
Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Proportional view



Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Computational complexity

	Random access	Insertion/ Extraction	<b>Sum of both</b>
Array	$O(1)$	$O(N)$	$O(N)$
List	$O(N)$	$O(1)$	$O(N)$
Super Tree	$O(\log(N))$	$O(\log(N))$	$O(\log(N))$

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Computational complexity (legend)

- $O(1)$  = constant 🤖
- $O(\log(N))$  = logarithmic 😊
- $O(N)$  = linear 😐
- $O(N \log(N))$  = “linearithmic” 😞
- $O(N^c)$  = polinomic 😡
- $O(c^N)$  = exponential 😈
- $O(N!)$  = factorial 🤯

**$N$** : size of the problem,     **$c$** : constant  $> 1$

Super Tree

Martín K.R.  
indizen

Intro

Super Tree










Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Computational complexity

	Random access	Insertion/ Extraction	<b>Sum of both</b>
Array			
List			
Super Tree			

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view







Applications

Similar  
proposals

Let's think  
about it

# Computational complexity

---

	(1 rand. access + 1 ins./extr.) $\times N = \mathbf{total}$	
Array		
List		
Super Tree		

---

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

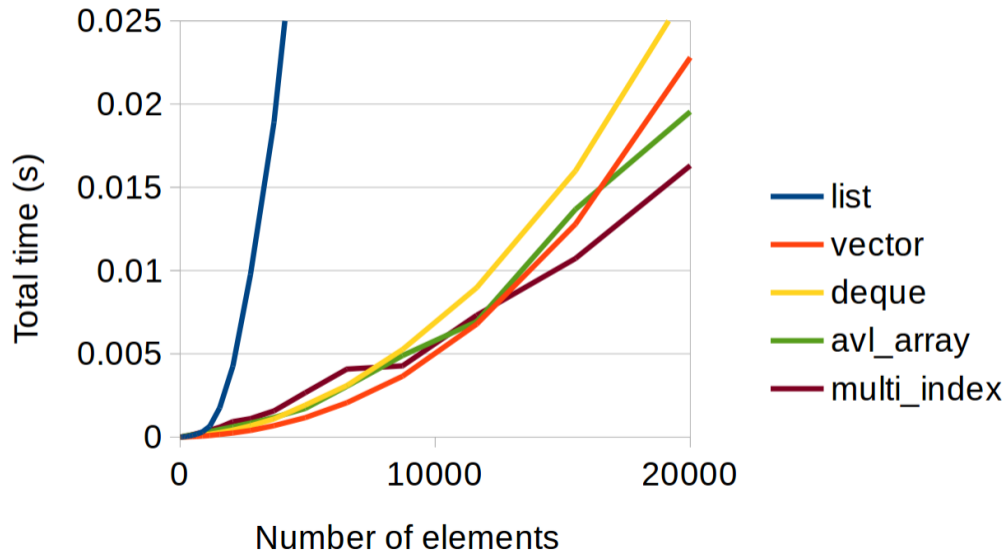
Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

## Results (1/3) — few elements



Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

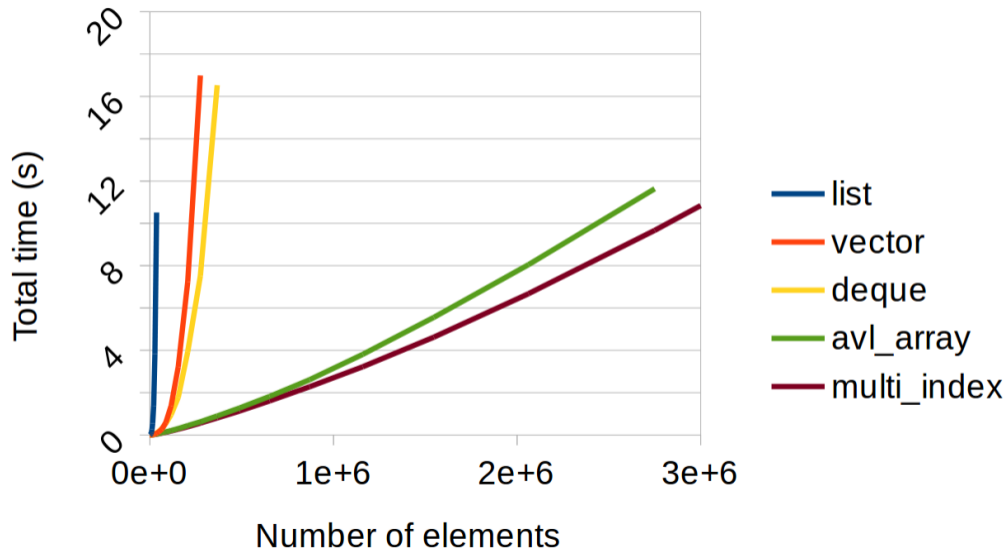
Applications

Similar  
proposals

Let's think  
about it



## Results (2/3) — many elements



Super Tree

Martín K.R.  
indizen

Intro

Super Tree

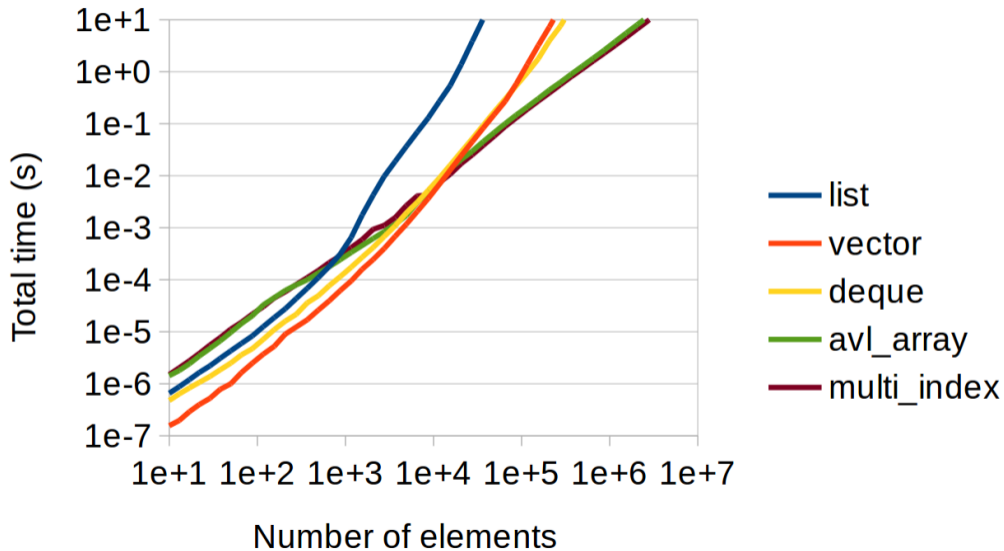
Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Results (3/3) — logarithmic scale



Super Tree

Martín K.R.  
indizen

Intro

Super Tree

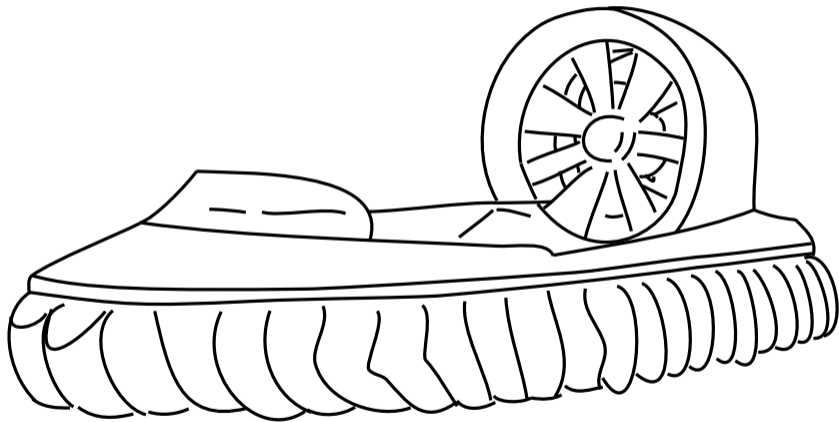
Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Ideal for the beach



Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Non proportional view

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

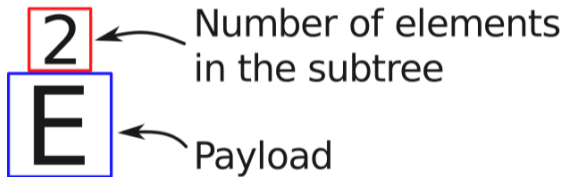
Applications

Similar  
proposals

Let's think  
about it

# Legend in proportional view

1 element = 1 unit



Super Tree

Martín K.R.  
indizen

Intro

Super Tree

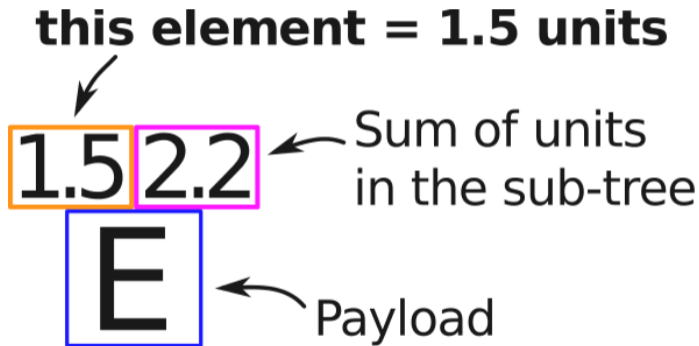
Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Legend in **non** proportional view



Super Tree

Martín K.R.  
indizen

Intro

Super Tree

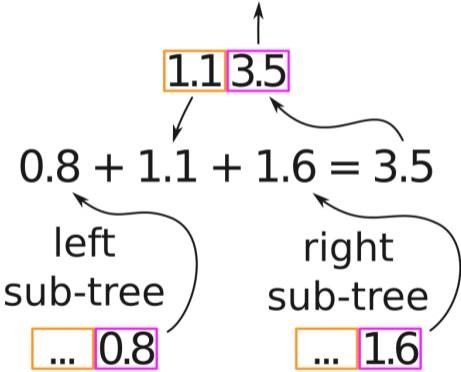
Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Sum in **non** proportional view



Super Tree

Martín K.R. indizen

Intro

Super Tree

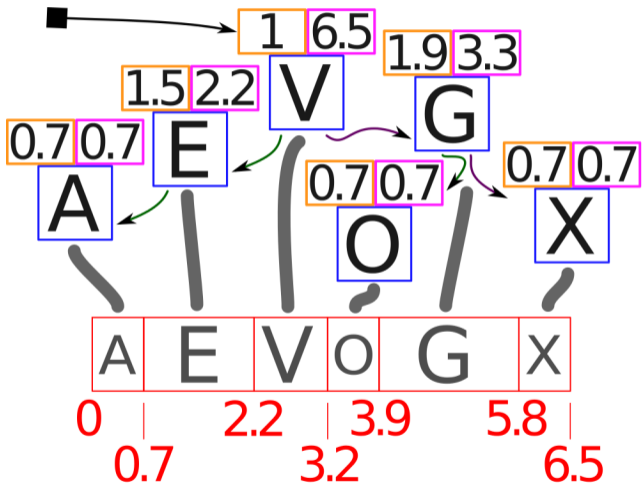
Non proportional view

Applications

Similar proposals

Let's think about it

# Non proportional view



Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it



# Applications

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Text editor

## Sequence of lines

- Number of bytes
- Number of lines after word wrap
- Number of characters
- If not plain text, number of pixels

gtk

“Ad hoc” B+ tree with number of characters and lines

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

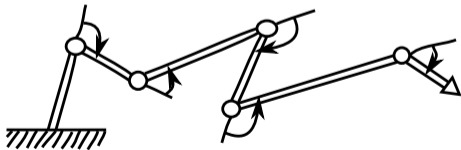
Applications

Similar  
proposals

Let's think  
about it

# Robot arm or chain of molecules

- Sequence of traslation and rotation transformations
- **Non** proportional view operation: matrix sum and product



Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Disk version: `shiftable_files`

- Implementation based on memory mapped files
- Horrible code (macros!)
- Metadata contained in the same file
- At closing time, choose:
  - 1 Recompact the file, or...
  - 2 leave it as is, with the metadata

## How to keep track of the sections?

Using an in-memory sequence with  
**non** proportional view

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Editing giant XML files

- A first pass can build an in-memory index (not necessarily complete)
- You can insert/extract nodes without rewriting the whole file
- You must keep the index updated, of course
- Recompile at closing?
  - 1 Yes: it becomes a normal XML again
  - 2 No: faster

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Similar proposals

Super Tree

Martín K.R.  
**indizen**

Intro

Super Tree

Non  
proportional  
view

Applications

**Similar  
proposals**

Let's think  
about it

# Multi Index (1/2)

```
boost::multi_index_container  
<  
    T,  
    boost::multi_index::indexed_by  
    <  
        boost::multi_index::ranked_non_unique  
        <  
            boost::multi_index::identity<T>,  
            unordered_less<T>  
        >  
    >  
>  
>
```

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Multi Index (2/2)

```
template<typename T>
struct unordered_less
{
    bool operator() (const T &,
                    const T &) const
    {
        return false;
    }
};
```

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it



## Similar proposals in Boost (1/2)

- 2004 – The oldest mention (I don't know if implemented), by Peter Palotas

<http://lists.boost.org/Archives/boost/2004/03/62823.php>

- 2006 – “Hierarchical Data Structures” by Bernhard Reiter and René Rivera

<http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2006/n2101.html#tr.hierarchy.augment>

- 2006 – “AVL Array” (horrible name, I know)

<http://sourceforge.net/projects/avl-array>

“Rank List” after debate in Boost forum

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Similar proposals in Boost (2/2)

- 2012 – Countertree by Vadim Stadnik  
[http://dl.dropbox.com/u/8437476/works/  
countertree/doc/index.html](http://dl.dropbox.com/u/8437476/works/countertree/doc/index.html) (broken link)
- 2015 – SegmentedTree by Chris Clearwater  
[https://det.github.io/segmented\\_tree/](https://det.github.io/segmented_tree/)

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Similar proposals not in Boost

- “Simon Tatham’s Algorithms Page”

<https://www.chiark.greenend.org.uk/>

`~sgtatham/algorithms/cbtree.html`

“Counted B-trees: An enhancement to the well known B-tree algorithms to allow you to **look up items in the tree by numeric index**, or to **find the numeric index of an item**. Useful for finding percentiles, [...]”

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let’s think  
about it

# Similar proposals in Python

- <https://pypi.python.org/pypi/rbtree>
- <https://pypi.python.org/pypi/pyavl>
- <https://pypi.python.org/pypi/blist>

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

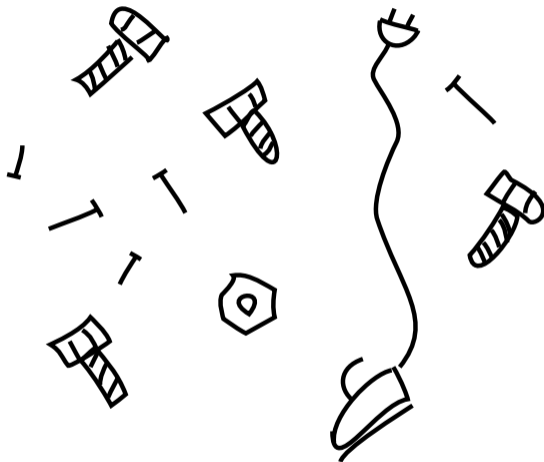
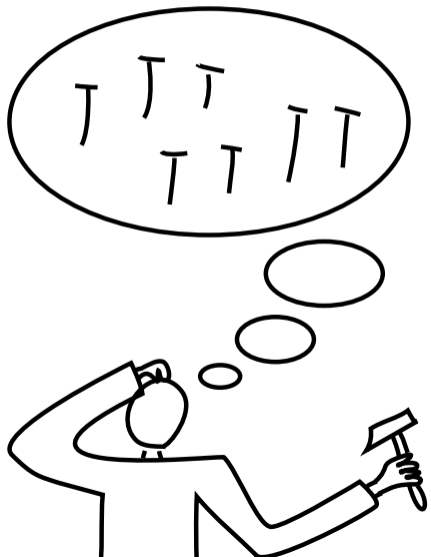
Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

# Let's think about it



Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it

Thanks a lot  
Any questions?

Super Tree

Martín K.R.  
indizen

Intro

Super Tree

Non  
proportional  
view

Applications

Similar  
proposals

Let's think  
about it