INSTITUT FÜR INFORMATIK
Lehrstuhl Prof. Zenger

Exploiting Maximum Parallelism in Hierarchical Numerical Algorithms

Alexander Pfaffinger

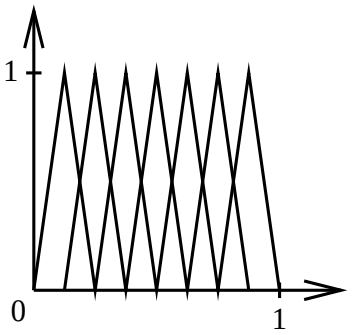
Contents

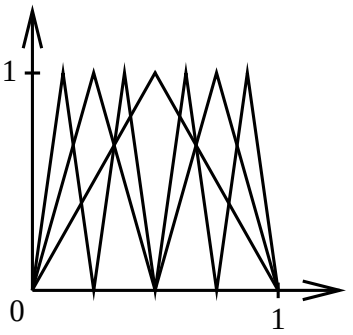
1. Hierarchical Bases
2. Sparse Grids
3. Algorithmic Structure
4. Imperative Description ?
5. Wrapper Streams
6. FASAN

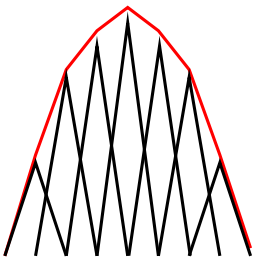
Hierarchical Bases

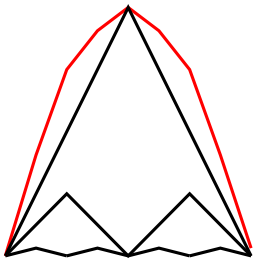
Hierarchical number representation: 3.1415927

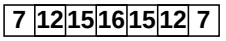
Function representation with hierarchical bases:

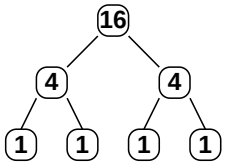






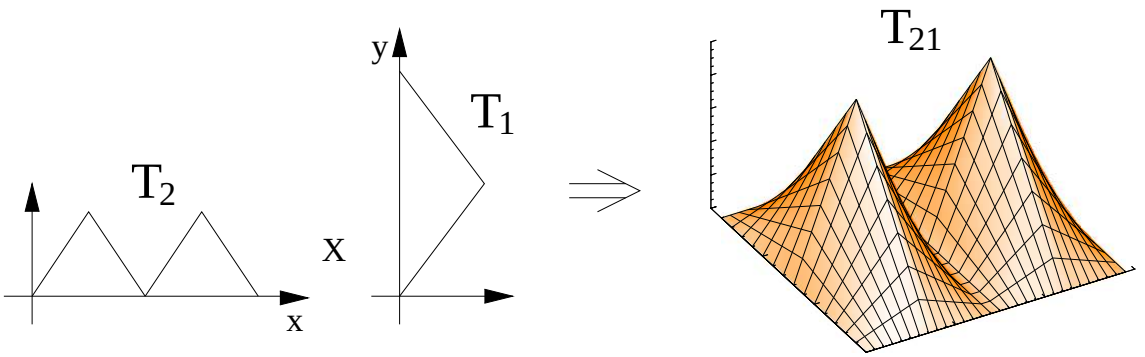






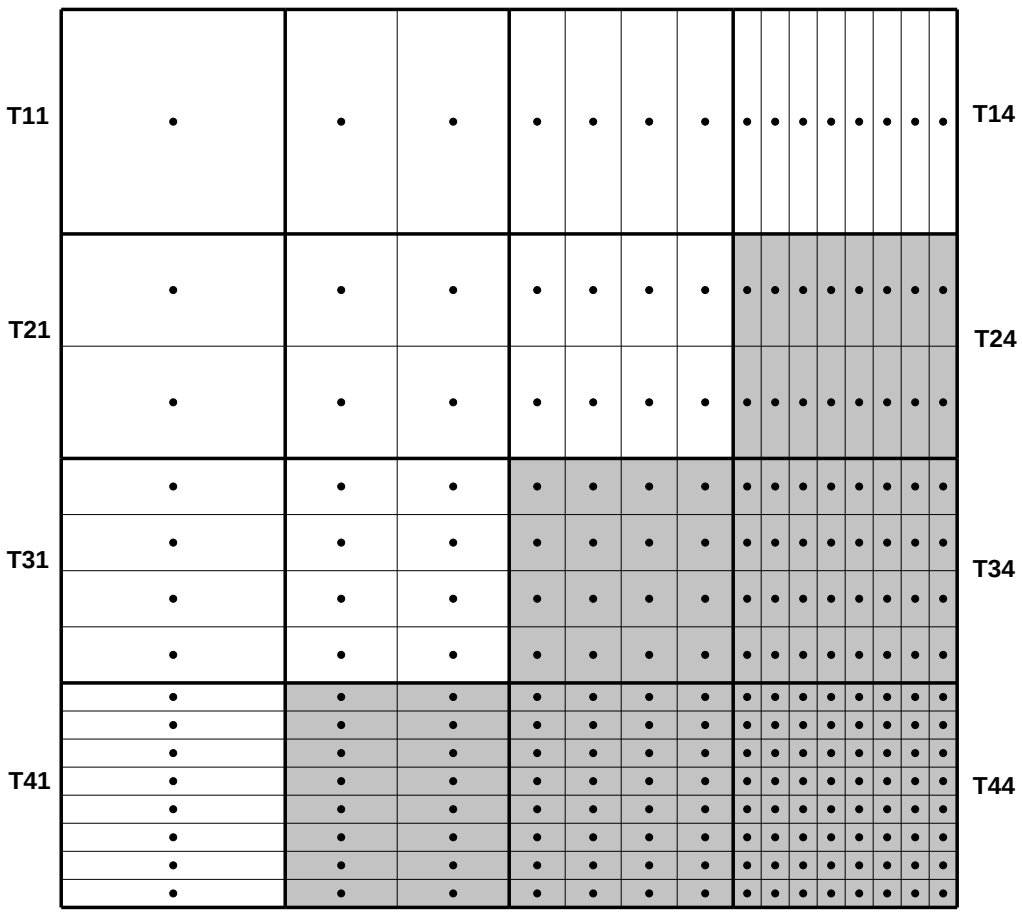
Generalization in Higher Dimensions

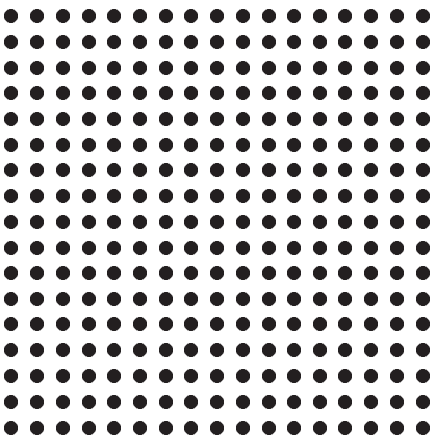
Product of 1-dimensional hierarchical basis functions:

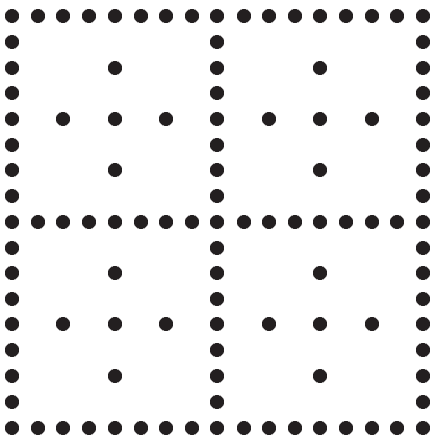


$$u_{i_1,\ldots,i_D}(x_1,\ldots,x_D) := \prod_{j=1}^{D} u_{i_j}(x_j)$$

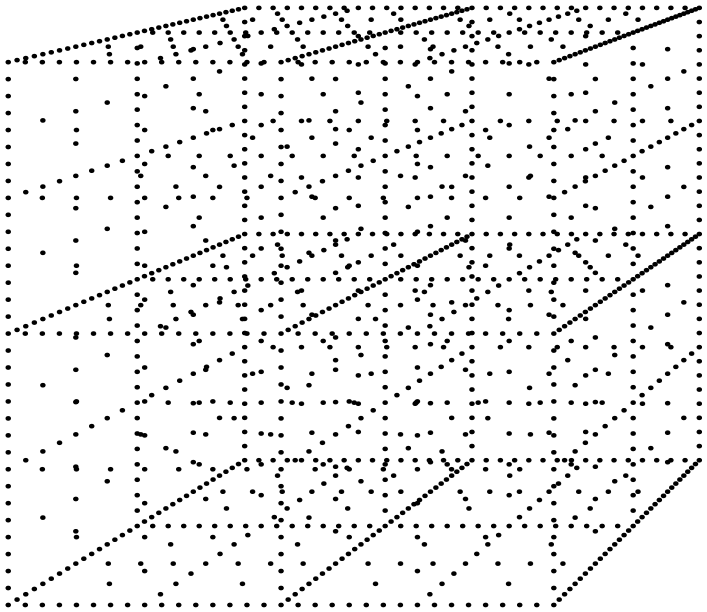
Sparse Grids







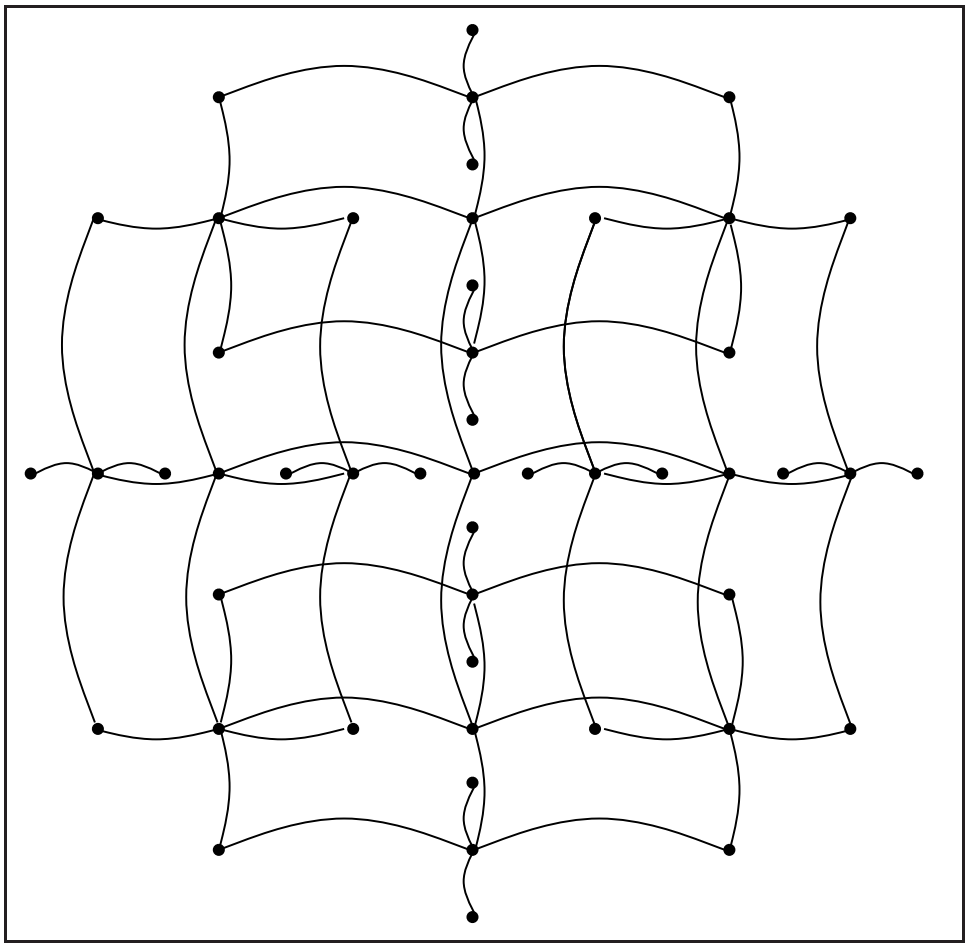
Comparison



	Full Grid	Sparse Grid
# Points $D = 3,\ N = 2^{10}$	N^D $1.07 \cdot 10^9$	$N(\log N)^{D-1}$ $1.02 \cdot 10^5$
Approximation Error	N^{-2}	$N^{-2}(\log N)^{D-1}$

But: more interdependence

Dependence Graph



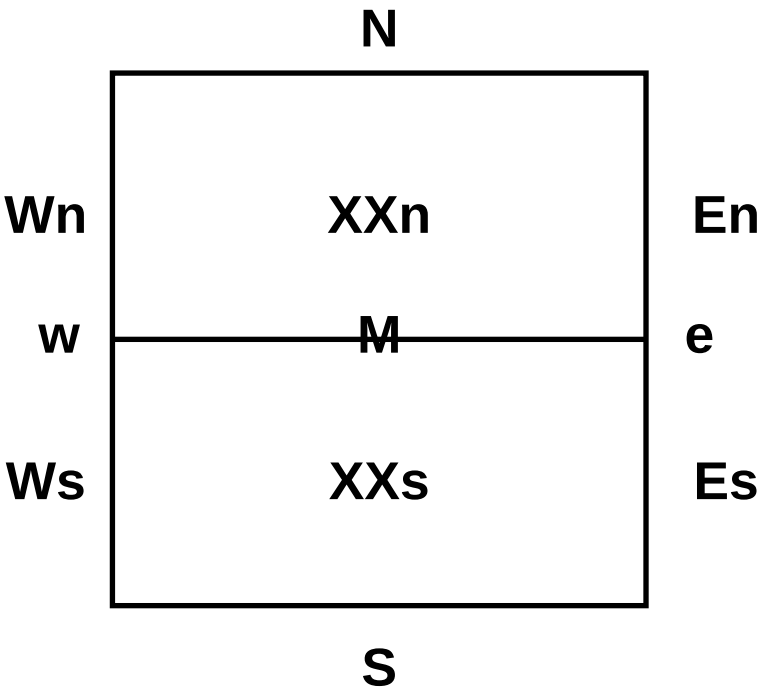
(Product graph of binary trees)

Algorithm (1D)

```
down (X,w,e): Y
begin
   ⟨Xw,x,Xe⟩ := X;
   m := f(w,e,x);
   if isNil(Xw) then
      Yw := nil;
      Ye := nil;
   else
      Yw := down (Xw,w,m) on host1;
      Ye := down (Xe,m,e) on host2;
   fi
   Y := ⟨Yw,m,Ye⟩
end
```

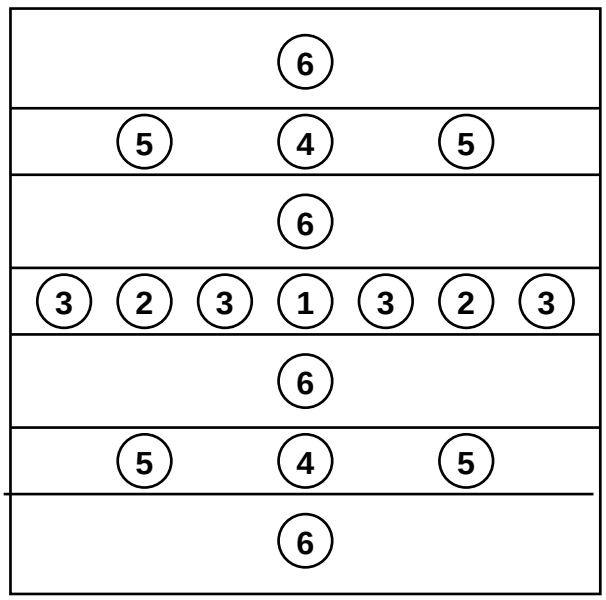
Parallel Complexity: $O(t) = O(\log(n))$

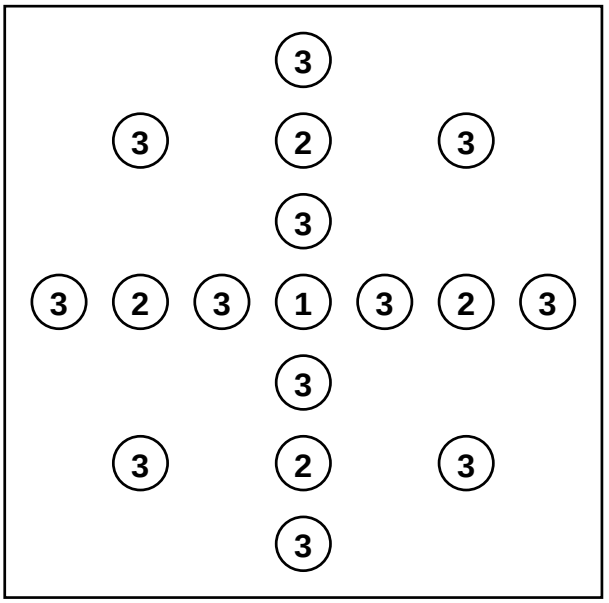
Algorithm (2D)




```
down2D (XX,N,S,W,E): YY
begin
   ⟨XXn,X,XXs⟩ := XX;
   ⟨Wn,w,Ws⟩ := W;
   ⟨En,e,Es⟩ := E;
   M := down1D (X,N,S,w,e);
   if isNil(XXn) then
      YYn := nil;
      YYs := nil;
   else
      YYn := down2D (XXn,N,M,Wn,En) on host1;
      YYs := down2D (XXs,M,S,Ws,Es) on host2;
   fi
   YY := ⟨YYn,M,YYs⟩
end
```

Complexity





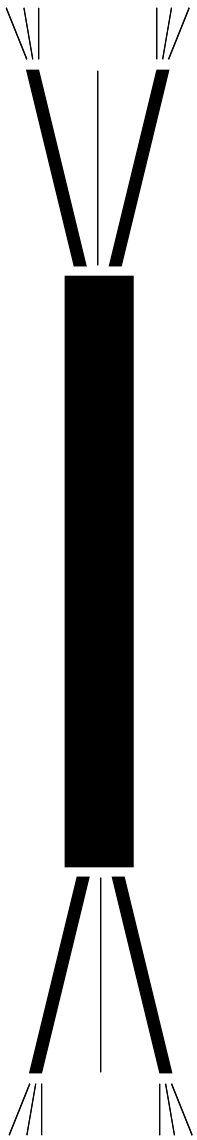
Parallel Complexity: $O(t^d)$
Optimal Complexity: $O(t)$

Problem: Optimal Complexity and Intuitive Description

Solution: Data flow semantics, Wrapper streams

Wrapper Streams

combine multiple streams into a single stream



recursive declaration:

width Tree = Tree ** 1 ** Tree

construction: $T := \langle Tl, t, Tr \rangle$

split: $\langle Tl, t, Tr \rangle := T$

functional syntax:

Streams are encapsulated and reached to the root. Recursive decomposition at descent.

efficient implementation:

Data flows directly and asynchronously from sender to consumer.

FASAN – Concepts

- program = textual representation of recursive data flow graph
- functions und streams (directed channels)
- elementary and composed functions
- initial composed function, recursive unfolding at runtime
- data driven semantics
- pipelining
- variable unfolding depth, sequential evaluation
- saving and loading of data flow graphs
- automatic migration und communication via PVM
- load distribution with location functions