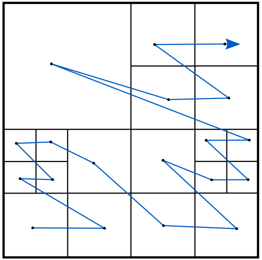
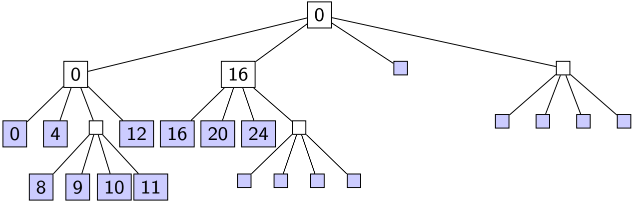
On the Theory of Discrete, Adaptive Space Filling Curves

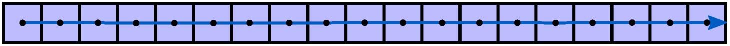
Johannes Holke (German Aerospace Center DLR, Cologne)
Carsten Burstedde (University of Bonn)
David Knapp (Uni Bonn and DLR)

SIAM Conference on Parallel Processing 2020

Intro







Intro

In this talk, we want to define a proper mathematical framework for SFCs in AMR.

Intro

In this talk, we want to define a proper mathematical framework for SFCs in AMR.

But wait: Don't we already have definitions?

Intro

In this talk, we want to define a proper mathematical framework for SFCs in AMR.

But wait: Don't we already have definitions?

Yes, but...

Intro

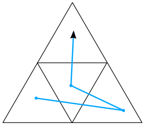
Most definitions of SFCs have at least one of these issues:

- Come from an analytical point of view
- Are not explicit
- Only on uniform subdivisions
- Actually allow stuff that should be illegal
- Restrict us to $1 : 2^d$ refinement

Intro

Most definitions of SFCs have at least one of these issues:

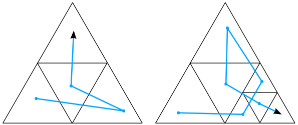
- Come from an analytical point of view
- Are not explicit
- Only on uniform subdivisions
- Actually allow stuff that should be illegal
- Restrict us to $1 : 2^d$ refinement



Intro

Most definitions of SFCs have at least one of these issues:

- Come from an analytical point of view
- Are not explicit
- Only on uniform subdivisions
- Actually allow stuff that should be illegal
- Restrict us to $1 : 2^d$ refinement



Intro

Theorem

Let X be an SFC, then Y holds.

Theorem

Let X be an SFC, then Y holds.

We need a <u>rigorous</u> definition of X to correctly proof the theorem.

Intro

We want a standalone definition suited for AMR, that is:

Intro

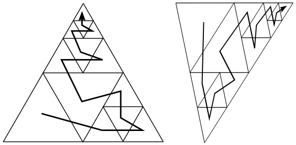
We want a standalone definition suited for AMR, that is:

- Independent of analytical SFC
-
-
-
-
-
-

Intro

We want a standalone definition suited for AMR, that is:

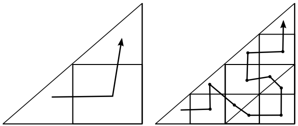
- Independent of analytical SFC
- Independent of geometry
-
-
-
-
-



Intro

We want a standalone definition suited for AMR, that is:

- Independent of analytical SFC
- Independent of geometry
- Allows for 'crazy' constructions
-
-
-
-



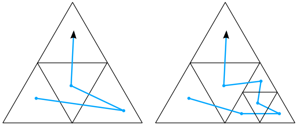
We want a standalone definition suited for AMR, that is:

- Independent of analytical SFC
- Independent of geometry
- Allows for 'crazy' constructions
- Utilizes trees/forests
-
-
-

Intro

We want a standalone definition suited for AMR, that is:

- Independent of analytical SFC
- Independent of geometry
- Allows for 'crazy' constructions
- Utilizes trees/forests
- Takes refinement hierarchy into account
-
-



Intro

We want a standalone definition suited for AMR, that is:

- Independent of analytical SFC
- Independent of geometry
- Allows for 'crazy' constructions
- Utilizes trees/forests
- Takes refinement hierarchy into account
- Allows us to construct new SFCs. See cross product
-

Intro

We want a standalone definition suited for AMR, that is:

- Independent of analytical SFC
- Independent of geometry
- Allows for 'crazy' constructions
- Utilizes trees/forests
- Takes refinement hierarchy into account
- Allows us to construct new SFCs. See cross product
- Allows for Code generation

Disclaimer

There exist lots of approaches where discrete SFCs are explicitely or implicitely described.
Not everything we present here is entirely new.
But maybe its the first time, we do it rigorously.

Idea

What is an SFC?

Idea

What is an SFC?

Some ordering of our elements.

$$\text{AMR mesh} \xrightarrow{\text{SFC}} \{0, 1, 2, 3, \ldots\} \tag{1}$$

Idea

What is an SFC?

Some ordering of our elements.

$$\text{AMR mesh} \xrightarrow{\text{SFC}} \{0, 1, 2, 3, \dots\} \tag{1}$$

Adaptive meshes are not alone, they come with a refinement hierarchy!

$$
\begin{tikzcd}
\text{AMR hierarchy} & \\
\text{AMR mesh} \arrow[u, "\subseteq"] \arrow[r, "\text{SFC}"] & \{0, 1, 2, 3, \ldots\}
\end{tikzcd}
\tag{2}
$$

$$
\begin{tikzcd}
\text{AMR hierarchy} \arrow[drr, "\text{SFC index}"] & & \\
\text{AMR mesh} \arrow[u, "\subseteq"] \arrow[rr, "\text{SFC}"] & & \{0, 1, 2, 3, \ldots\}
\end{tikzcd}
\tag{2}
$$

$$
\begin{tikzcd}
\text{Refinement space} \arrow[dr, "\text{SFC index}"] & \\
\text{Refinement} \arrow[u, "\cup|"] \arrow[r, "\text{SFC}"] & \{\, 0, 1, 2, 3, \dots \,\}
\end{tikzcd}
\tag{2}
$$

Refinement spaces

Definition (refinement space)

A **refinement space** consists of:

- A set $\mathcal{S}$, the **elements**
- A map $\ell\colon \mathcal{S} \to \mathbb{N}_0$, the **level**
- Maps $R^l\colon \ell^{-1}(l) =: \mathcal{S}^l \to \mathcal{P}(S^{l+1})$, the **refinement maps**

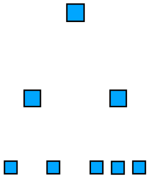
such that

Definition (refinement space)

A **refinement space** consists of:

- A set $\mathcal{S}$, the **elements**
- A map $\ell\colon \mathcal{S} \to \mathbb{N}_0$, the **level**
- Maps $R^l\colon \ell^{-1}(l) =: \mathcal{S}^l \to \mathcal{P}(S^{l+1})$, the **refinement maps**

such that



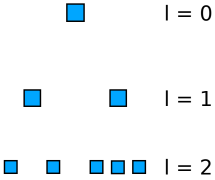
Refinement spaces

Definition (refinement space)

A **refinement space** consists of:

- A set $\mathcal{S}$, the **elements**
- A map $\ell\colon \mathcal{S} \to \mathbb{N}_0$, the **level**
- Maps $R^l\colon \ell^{-1}(l) =: \mathcal{S}^l \to \mathcal{P}(S^{l+1})$, the **refinement maps**

such that

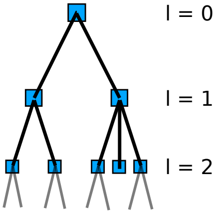


Definition (refinement space)

A **refinement space** consists of:

- A set $\mathcal{S}$, the **elements**
- A map $\ell\colon \mathcal{S} \to \mathbb{N}_0$, the **level**
- Maps $R^l\colon \ell^{-1}(l) =: \mathcal{S}^l \to \mathcal{P}(S^{l+1})$, the **refinement maps**

such that



Definition (refinement space)

A **refinement space** consists of:

- A set $\mathcal{S}$, the **elements**
- A map $\ell\colon \mathcal{S} \to \mathbb{N}_0$, the **level**
- Maps $R^l\colon \ell^{-1}(l) =: \mathcal{S}^l \to \mathcal{P}(S^{l+1})$, the **refinement maps**

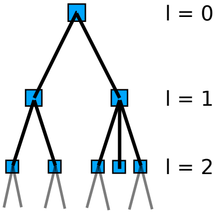
such that

- Every element of level $l > 0$ has a **unique parent**:

$$\bigcup_{E\in\mathcal{S}^l} R^l(E) = \mathcal{S}^{l+1}. \tag{3}$$

$$R^l(E) \cap R^l(E') = \emptyset \text{ for } E \neq E' \in \mathcal{S}^l \tag{4}$$

- $\mathcal{S}^0 = \{\, \mathcal{E}_0 \,\}$

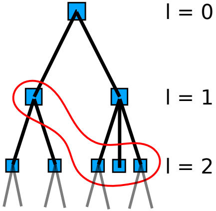


Refinements

Definition (refinement)

A **refinement** of a refinement space $\mathcal{S}$ is a subset $\mathscr{S}$ constructed via successivly refining level 0, thus:

- $\mathscr{S} = \mathcal{S}^0$ is a refinement.
- $\mathscr{S} \backslash E \cup R^l(E)$ is a refinement for each $E \in \mathscr{S}$.



Definition (SFC index)

An **SFC index** $\mathcal{I}$ is a map

$$\mathcal{I}\colon \mathcal{S} \to \mathbb{N}_0 \tag{5}$$

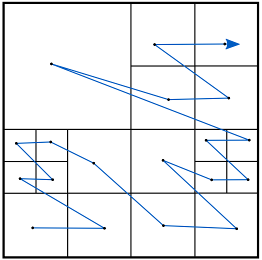
- $\mathcal{I} \times \ell\colon \mathcal{S} \to \mathbb{N}_0 \times \mathbb{N}_0$ is injective. **Restricted to a level, $\mathcal{I}$ is unique.**
- E ancestor of $E' \Rightarrow \mathcal{I}(E) \leq \mathcal{I}(E')$. **Refining does not decrease the index.**
- $\mathcal{I}(E) < \mathcal{I}(\hat{E})$ and $\hat{E}$ not a descendant of $E \Rightarrow \mathcal{I}(E) \leq \mathcal{I}(E') < \mathcal{I}(\hat{E})$ for all descendants E' of E. **Refining is 'local'.**

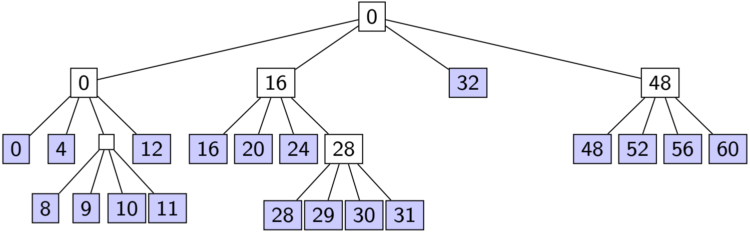
Definition (Space-filling curve)

A (discrete) **Space-filling curve** is an SFC index restricted to a refinement.

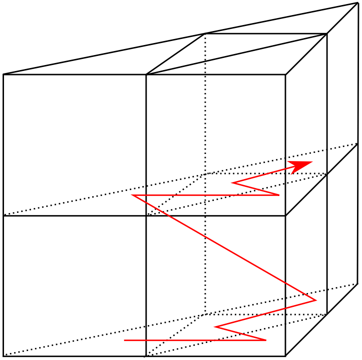
$$\mathcal{I}_{|\mathscr{S}} : \mathscr{S} \to \mathbb{N}_0 \tag{6}$$

Example

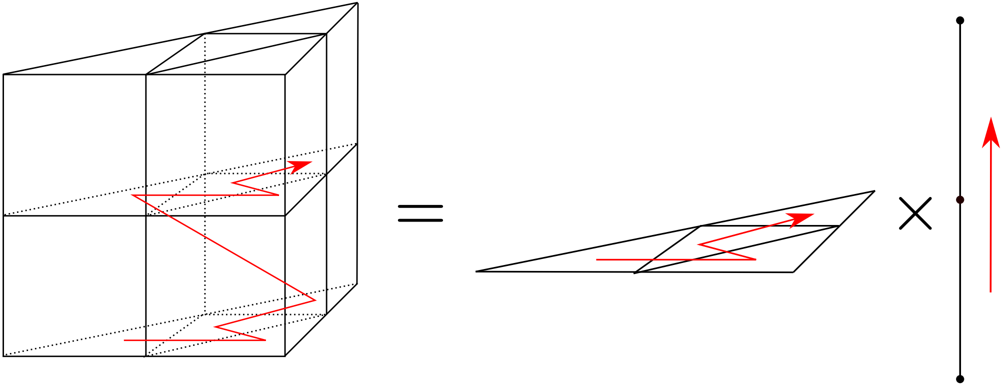




Cross product



Cross product



Definition (Cross product of refinement spaces)

Let $(\mathcal{S}_1, R_1, \ell_1)$ and $(\mathcal{S}_2, R_2, \ell_2)$ be refinement spaces. The cross product $\mathcal{S}_\times = \mathcal{S}_1 \times \mathcal{S}_2$ is defined via

- $\mathcal{S}^l_\times := \mathcal{S}^l_1 \times \mathcal{S}^l_2$
- $\ell_\times(E_1, E_2) := \ell(E_1) = \ell(E_2)$
- $R^l_\times(E_1, E_2) := R^l_1(E_1) \times R^l_2(E_2) \subset \mathcal{S}^{l+1}_1 \times \mathcal{S}^{l+1}_2 = \mathcal{S}^{l+1}_\times$

Proposition

The cross product is a refinement space.

$\square$

Cross product

Definition (Cross product of SFC indices)

The cross product index $\mathcal{I}_\times = \mathcal{I}_1 \times \mathcal{I}_2$ on $\mathcal{S}_1 \times S_2$:

$$\mathcal{I}_\times : \mathcal{S}_1 \times \mathcal{S}_2 \to \mathbb{N}_0 \tag{7}$$

defined recursively by

$$\mathcal{I}_\times(E_1, E_2) := \mathcal{I}_x(P_1, P_2) + m_1^{>l} m_2^{>l} \left(\mathrm{sibid}_1(E_1) * \#\mathrm{siblings}(E_2) + \mathrm{sibid}(E_2)\right) \tag{8}$$

Cross product

Definition (Cross product of SFC indices)

The cross product index $\mathcal{I}_\times = \mathcal{I}_1 \times \mathcal{I}_2$ on $\mathcal{S}_1 \times S_2$:

$$\mathcal{I}_\times : \mathcal{S}_1 \times \mathcal{S}_2 \to \mathbb{N}_0 \tag{7}$$

defined recursively by

$$\mathcal{I}_\times(E_1, E_2) := \mathcal{I}_x(P_1, P_2) + m_1^{>l} m_2^{>l} (\mathsf{sibid}_1(E_1) * \#\mathrm{siblings}(E_2) + \mathsf{sibid}(E_2)) \tag{8}$$

where

$$m^{>l} := \prod_{i>l} \max_E |R^i(E)| \tag{9}$$

Definition (Cross product of SFC indices)

The cross product index $\mathcal{I}_\times = \mathcal{I}_1 \times \mathcal{I}_2$ on $\mathcal{S}_1 \times S_2$:

$$\mathcal{I}_\times : \mathcal{S}_1 \times \mathcal{S}_2 \to \mathbb{N}_0 \tag{7}$$

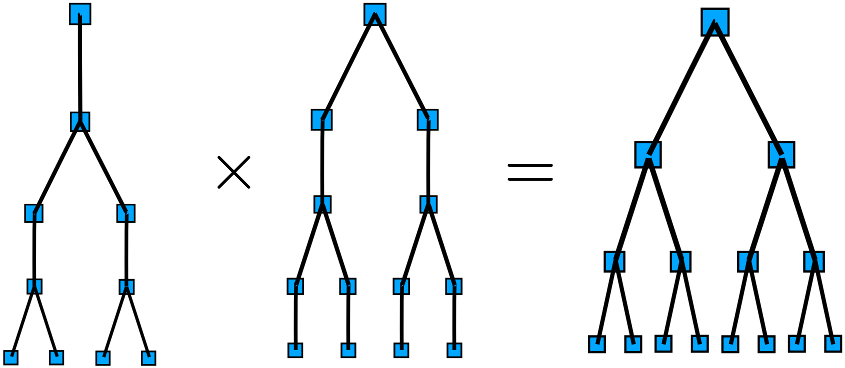
defined recursively by

$$\mathcal{I}_\times(E_1, E_2) := \mathcal{I}_x(P_1, P_2) + m_1^{>l} m_2^{>l} (\mathrm{sibid}_1(E_1) * \#\mathrm{siblings}(E_2) + \mathrm{sibid}(E_2)) \tag{8}$$

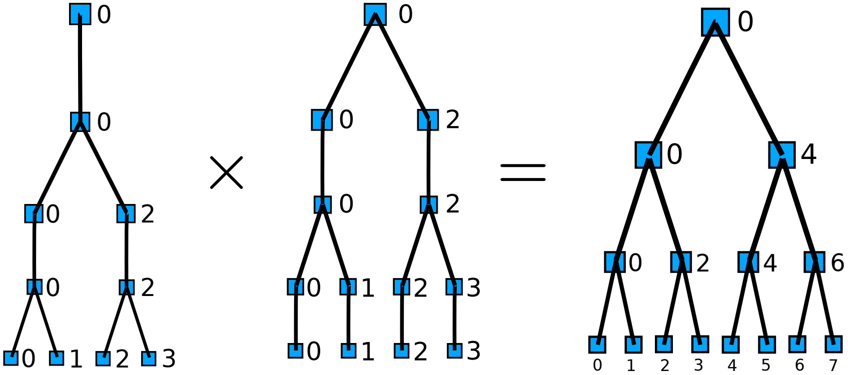
Proposition

The cross product index is an SFC index. □

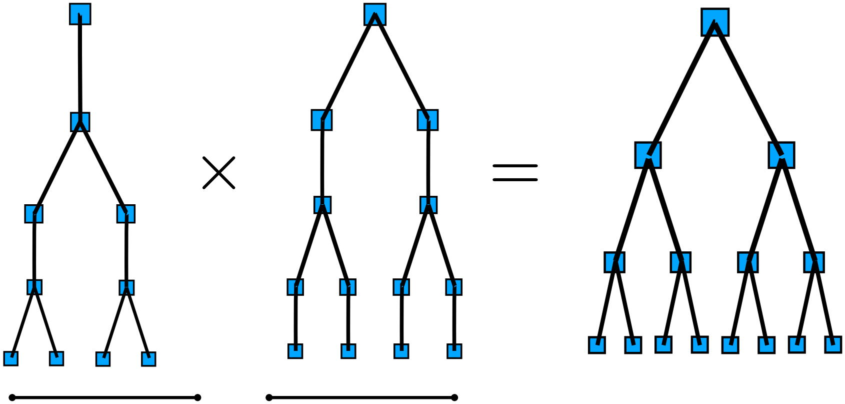
An interesting example



An interesting example

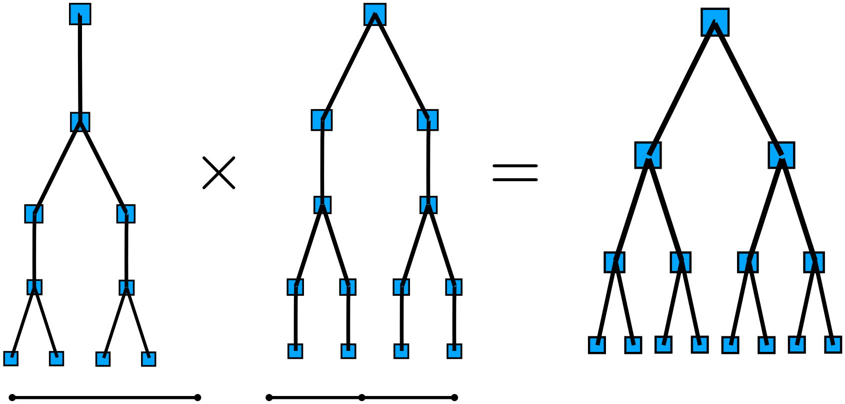


An interesting example



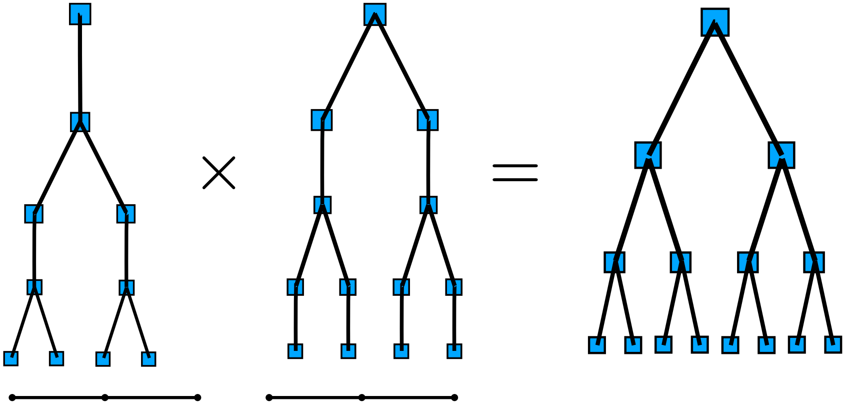
$\ell = 0$

An interesting example



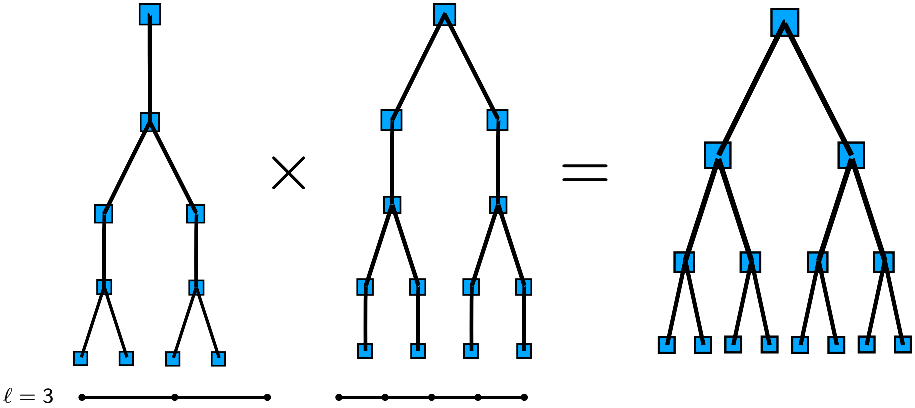
$\ell = 1$

An interesting example

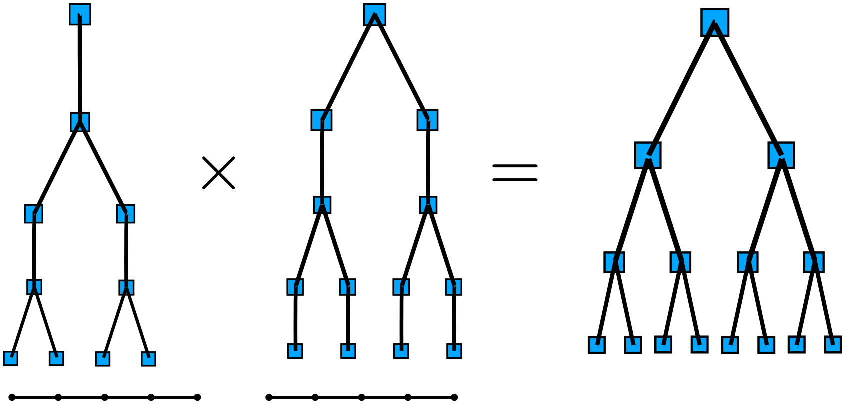


$\ell = 2$

An interesting example

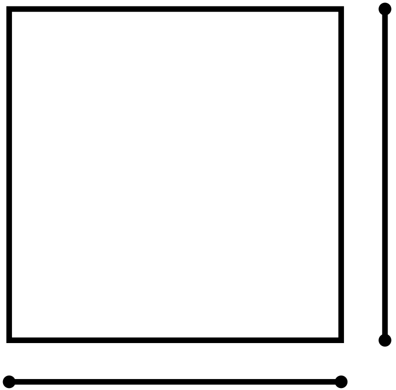


An interesting example



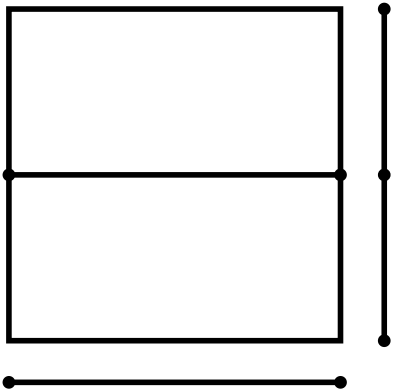
$\ell = 4$

An interesting example



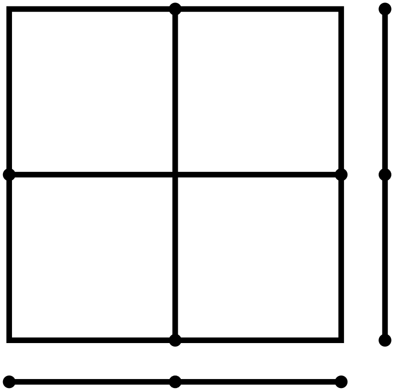
$$\ell = 0$$

An interesting example



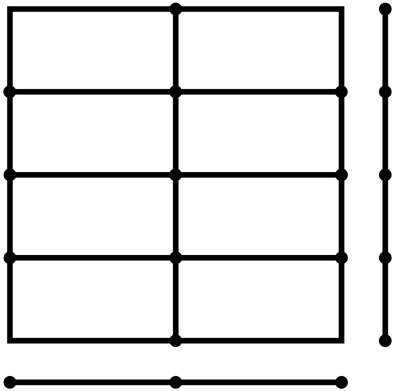
$$\ell = 1$$

An interesting example



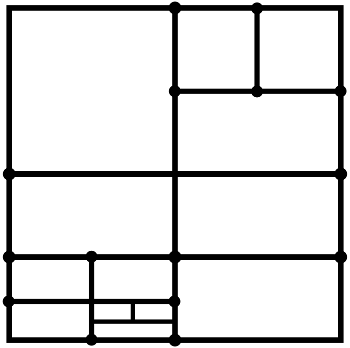
$$\ell = 2$$

An interesting example

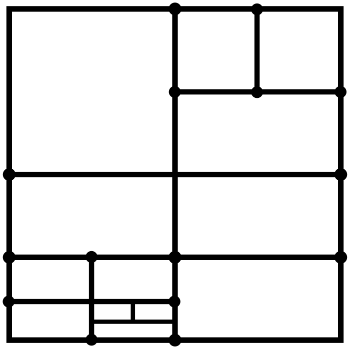


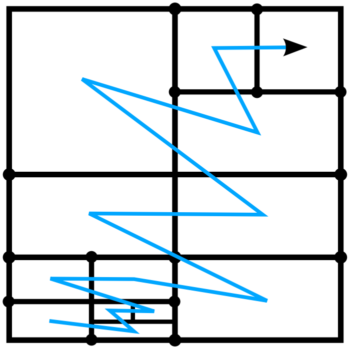
$\ell = 3$

An interesting example

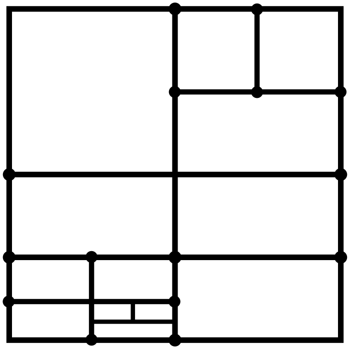


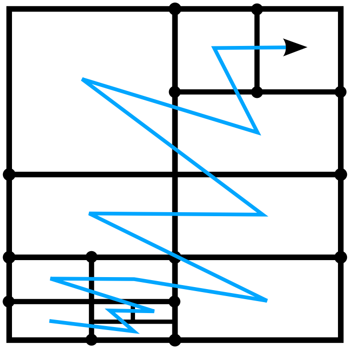
An interesting example





An interesting example





$\Rightarrow$ An SFC for dyadic refinement.

Furthermore

- Can be straightforwardly extended to forest
- Can define partitions
- TODO: How to encode (face-)neighbors?
- TODO: Can we find other cross products?

Furthermore

- Can be straightforwardly extended to forest
- Can define partitions
- TODO: How to encode (face-)neighbors?
- TODO: Can we find other cross products?

Thank you.

Questions?

Johannes Holke.
Scalable algorithms for parallel tree-based adaptive mesh refinement with general element types.
PhD thesis, Rheinische Friedrich-Wilhelms-Universität Bonn, 2018.

Johannes Holke, David Knapp, and Carsten Burstedde.
On the theory of discrete, adaptive space filling curves.
In preparation, 2020.

Johannes Holke.
Scalable algorithms for parallel tree-based adaptive mesh refinement with general element types.
PhD thesis, Rheinische Friedrich-Wilhelms-Universität Bonn, 2018.

Johannes Holke, David Knapp, and Carsten Burstedde.
On the theory of discrete, adaptive space filling curves.
In preparation, 2020.