Basis Computation Algorithms

Nina S. T. Hirata, Roberto Hirata Jr. and Junior Barrera

Department of Computer Science
University of São Paulo

ISMM 2007
8th International Symposium on Mathematical Morphology
October 11-13, 2007

Thanks to FAPESP and CNPq.

Outline

1. Morphological operator decomposition
   - W-operator definition
   - W-operator sup-decomposition
2. Basis computation
   - Binary operators
   - Binary input multiple output operators
   - Gray-scale operators
3. Concluding remarks

Outline

1. Morphological operator decomposition
   - W-operator definition
   - W-operator sup-decomposition
2. Basis computation
   - Binary operators
   - Binary input multiple output operators
   - Gray-scale operators
3. Concluding remarks

Outline

1. Morphological operator decomposition
   - W-operator definition
   - W-operator sup-decomposition
2. Basis computation
   - Binary operators
   - Binary input multiple output operators
   - Gray-scale operators
3. Concluding remarks

Binary W-operators (1)

Translation-invariance

S: binary image

S_z: S translated by z

$$[\Psi(S)]_z = \Psi(S_z)$$

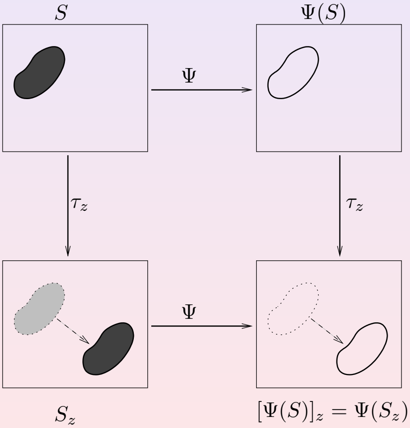
Binary W-operators (1)

Translation-invariance

S: binary image

S_z: S translated by z

$[\Psi(S)]_z = \Psi(S_z)$



Binary W-operators (2)

Local definition

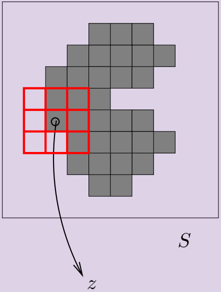
Ψ is **locally defined** within a window W iff

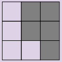
$$z \in \Psi(S) \iff z \in \Psi(S \cap W_z)$$

Binary *W*-operators (3)

$$\boxed{W\text{-operator} = \text{translation-invariance} + \text{local definition}}$$

Characterization by a function



$\Psi(S)(z) = \psi\Big($  $\Big)$

ψ is a Boolean function

Binary W-operator sup-decomposition

W-operator: $\Psi : \mathcal{P}(E) \rightarrow \mathcal{P}(E)$, $E = \mathbb{Z}^2$

$\psi : \mathcal{P}(W) \rightarrow \{0,1\}$ (or $\psi : \{0,1\}^n \rightarrow \{0,1\}$, $n = |W|$)

Kernel and basis

- Kernel: $\mathcal{K}(\Psi) = \{X \in \mathcal{P}(W) : \psi(X) = 1\}$
- Basis: $\mathbf{B}(\Psi)$, maximal intervals in $\mathcal{K}(\Psi)$

Binary W-operator sup-decomposition

W-operator: $\Psi : \mathcal{P}(E) \to \mathcal{P}(E)$, $E = \mathbb{Z}^2$

$\psi : \mathcal{P}(W) \to \{0, 1\}$ (or $\psi : \{0, 1\}^n \to \{0, 1\}$, $n = |W|$)

Kernel and basis

- Kernel: $\mathcal{K}(\Psi) = \{X \in \mathcal{P}(W) : \psi(X) = 1\}$
- Basis: $\mathbf{B}(\Psi)$, maximal intervals in $\mathcal{K}(\Psi)$

Binary W-operator sup-decomposition

W-operator: $\Psi : \mathcal{P}(E) \to \mathcal{P}(E)$, $E = \mathbb{Z}^2$

$\psi : \mathcal{P}(W) \to \{0, 1\}$ (or $\psi : \{0, 1\}^n \to \{0, 1\}$, $n = |W|$)

Kernel and basis

- Kernel: $\mathcal{K}(\Psi) = \{X \in \mathcal{P}(W) : \psi(X) = 1\}$
- Basis: $\mathbf{B}(\Psi)$, maximal intervals in $\mathcal{K}(\Psi)$

Binary W-operator sup-decomposition

Canonical sup-decompositions

- Interval operator: $\lambda_{[A,B]}(X) = 1 \iff X \in [A, B]$
- Kernel decomposition:
  $\psi(X) = max\{\lambda_{[A,A]}(X) : [A, A] \subseteq \mathcal{K}(\psi)\}$
- Basis decomposition:
  $\psi(X) = max\{\lambda_{[A,B]}(X) : [A, B] \in \mathbf{B}(\psi)\}$

Binary W-operator sup-decomposition

Canonical sup-decompositions

- Interval operator: $\lambda_{[A,B]}(X) = 1 \iff X \in [A, B]$
- Kernel decomposition:
  $\psi(X) = max\{\lambda_{[A,A]}(X) : [A, A] \subseteq \mathcal{K}(\psi)\}$
- Basis decomposition:
  $\psi(X) = max\{\lambda_{[A,B]}(X) : [A, B] \in \mathbf{B}(\psi)\}$

Binary W-operator sup-decomposition

Canonical sup-decompositions

- Interval operator: $\lambda_{[A,B]}(X) = 1 \iff X \in [A, B]$
- Kernel decomposition:
  $\psi(X) = max\{\lambda_{[A,A]}(X) : [A, A] \subseteq \mathcal{K}(\psi)\}$
- Basis decomposition:
  $\psi(X) = max\{\lambda_{[A,B]}(X) : [A, B] \in \mathbf{B}(\psi)\}$

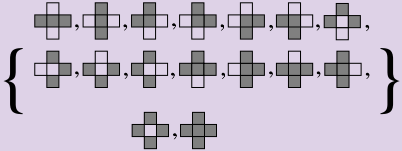
Binary W-operator sup-decomposition

Canonical sup-decompositions

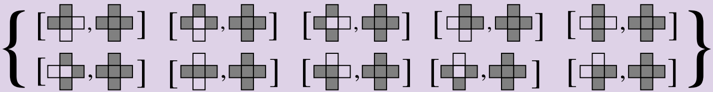
- Interval operator: $\lambda_{[A,B]}(X) = 1 \iff X \in [A, B]$
- Kernel decomposition:
  $\psi(X) = max\{\lambda_{[A,A]}(X) : [A, A] \subseteq \mathcal{K}(\psi)\}$
- Basis decomposition:
  $\psi(X) = max\{\lambda_{[A,B]}(X) : [A, B] \in \mathbf{B}(\psi)\}$

Binary *W*-operator: example of kernel and basis

Kernel of the median filter (5-point cross-window)

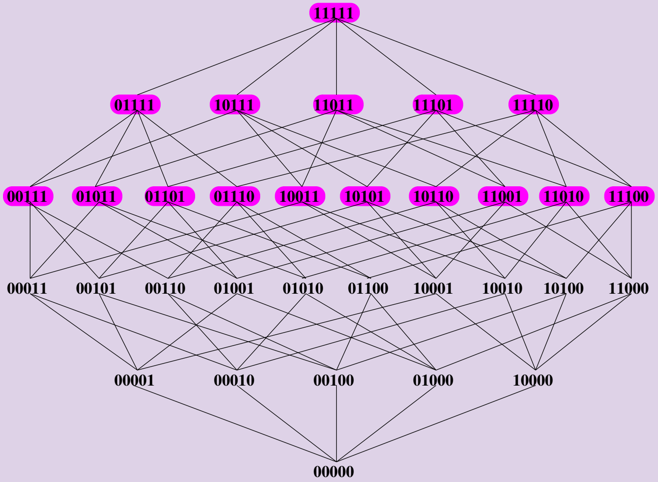


Basis of the median filter (5-point cross-window)



Binary W-operator: example of kernel and basis

Lattice view: kernel of the median filter



W-operators

$K = \{0, 1, \ldots, k\}$
K^E: collection of images defined on E, with $k+1$ grey-levels

Similar definitions exist for

- binary input multi-level output operators

$$\Psi : \mathcal{P}(E) \rightarrow K^E$$

  characterized by functions $\psi : \{0, 1\}^n \rightarrow K$

- multi-level input multi-level output (gray-level) operators

$$\Psi : K^E \rightarrow K^E$$

  characterized by functions $\psi : K^n \rightarrow K$

Basis computation

Problem definition

Given a set of elements known to be in the kernel of an operator and others known not to be in the kernel, find a minimal set of intervals that corresponds to the given elements.

For the binary case, this problem corresponds to the minimization of incompletely specified Boolean functions.

Basis computation – binary W-operators

Incremental splitting of intervals

- Start from the whole lattice
- Successively remove elements X such that $\psi(X) = 0$ from the lattice.
- At each removing step, represent the remaining elements of the lattice by means of a minimal set of maximal intervals that cover them.
- The resulting intervals, after finishing the removing, cover only elements such that $\psi(X) = 1$ (and eventually some don't cares).

Basis computation – binary W-operators

Interval splitting rule

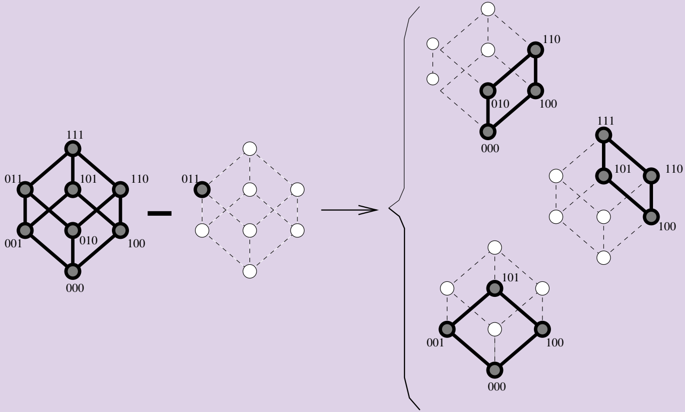
- How to express the remaining elements in an interval (after removing some of them) as a set of maximal sub-intervals ?

  Let $[A, B]$ in $[\emptyset, W]$ and $X \in [A, B]$. The set of maximal intervals contained in $[A, B] \setminus \{X\}$ is given by

  $$\{[A, B \cap \{a\}^c] : a \in X \cap A^c\} \cup \{[A \cup \{b\}, B] : b \in B \cap X^c\}$$

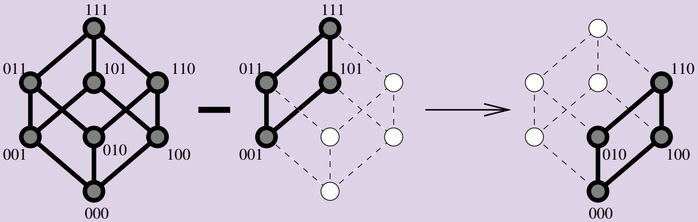
Basis computation – binary W-operators

Removing of 011 from $\{0,1\}^3$



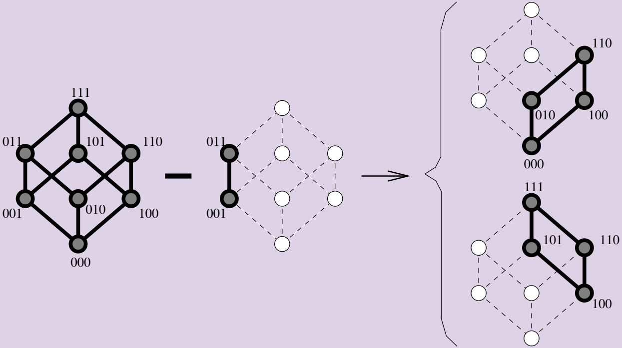
Basis computation – binary W-operators

Removing of $[001, 111]$ from $\{0, 1\}^3$



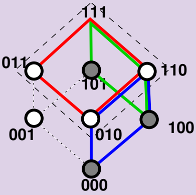
Basis computation – binary W-operators

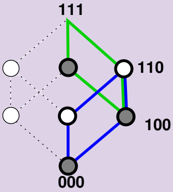
Removing of $[001, 011]$ from $\{0,1\}^3$



Basis computation – binary W-operators

Don't cares imply that some intervals may be eliminated during the splitting process





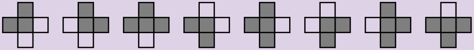
Left: three intervals after removing of 001

Right: interval $[010, 111]$ not needed (it covers only don't cares)

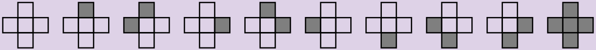
Binary W-operators – Example

Kernel and basis example

Elements known to be in the kernel ($\psi(X) = 1$)



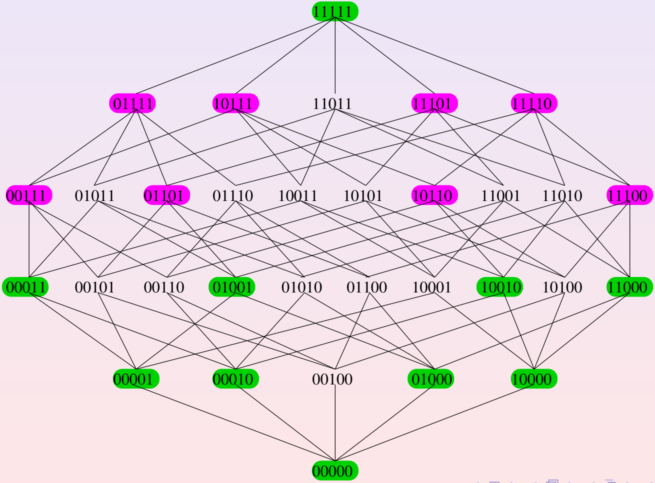
Elements known not to be in the kernel ($\psi(X) = 0$)



Basis (consistent with the above elements) is given by the set of intervals

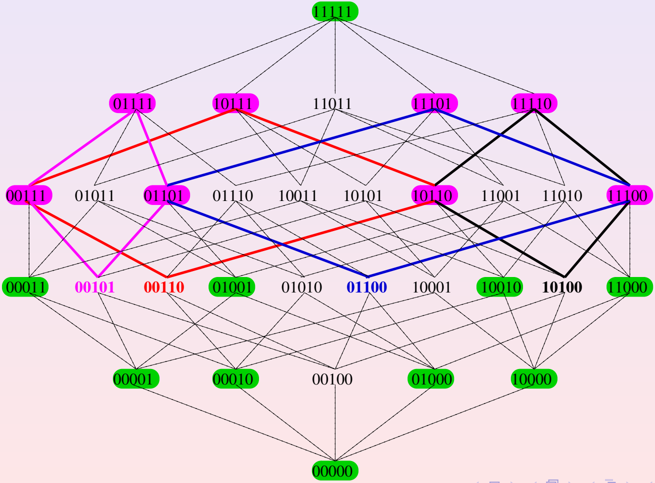
Binary W-operators – Example

Lattice view of the kernel and non-kernel elements



Binary W-operators – Example

Lattice view of the basis



Basis computation – binary-input multiple-output operators

Definition

Mappings of the type

$$\psi : \{0,1\}^n \rightarrow K$$

Example of such operator: character classification

Binary-input multiple-output operators

Kernel

Kernel of ψ at level i, $i = 0, 1, \ldots, k$

$$\mathbf{x} \in \mathcal{K}_i(\psi) \Longleftrightarrow \psi(\mathbf{x}) \geq i, \quad \mathbf{x} \in \{0,1\}^n$$

$\mathcal{K}_k(\psi) \subseteq \mathcal{K}_{k-1}(\psi) \subseteq \cdots \subseteq \mathcal{K}_1(\psi) \subseteq \mathcal{K}_0(\psi)$

Basis

Basis of ψ at level i

$\mathbf{B}_i(\psi)$: the collection of maximal intervals in $\mathcal{K}_i(\psi)$

Binary input multiple output operators

kernel and basis sup-decomposition

Sup-decomposition of ψ in terms of its kernel and basis

$$\psi(\mathbf{x}) = max\{i : \mathbf{x} \in \mathcal{K}_i(\psi), i = 0, 1, \ldots, k\},$$

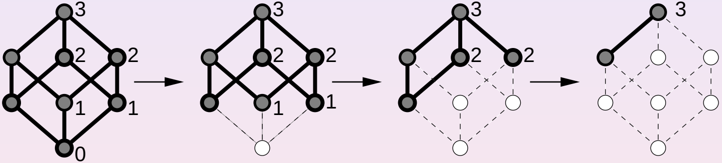
$$\psi(\mathbf{x}) = max\{i : \mathbf{x} \in [\mathbf{a}, \mathbf{b}], [\mathbf{a}, \mathbf{b}] \in \mathbf{B}_i(\psi), i = 0, 1, \ldots, k\}\,.$$

Basis computation – binary-input multiple-output operators

Algorithm

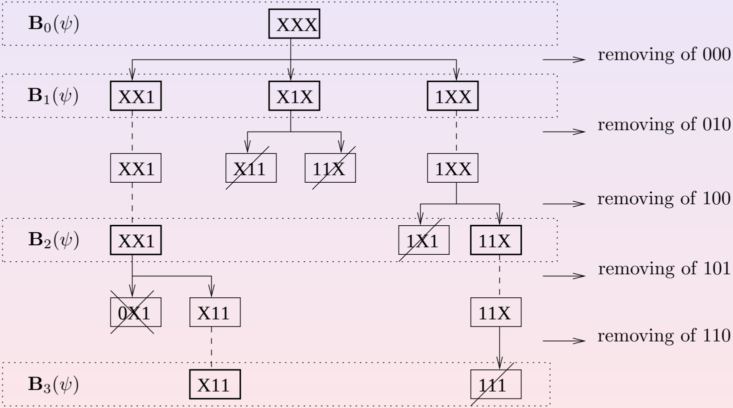
- Start from the whole lattice
- Remove successively all elements such that $\psi(X) = 0$; this results in $\mathbf{B}_1(\psi)$
- From intervals in $\mathbf{B}_1(\psi)$, remove successively all elements such that $\psi(X) = 1$; this results in $\mathbf{B}_2(\psi)$
- From intervals in $\mathbf{B}_2(\psi)$, remove successively all elements such that $\psi(X) = 2$; this results in $\mathbf{B}_3(\psi)$
- and so on, until all elements such that $\psi(X) = k - 1$ are removed

Binary input multiple output operators



$\mathbf{B}_0(\psi)$ $\mathbf{B}_1(\psi)$ $\mathbf{B}_2(\psi)$ $\mathbf{B}_3(\psi)$

Binary input multiple output operators



Gray-scale operators

Definition

Mappings of the type

$$\psi : K^n \longrightarrow K$$

Gray-scale operators

Kernel

Kernel of ψ at level i, $i = 0, 1, \ldots, k$

$$\mathbf{x} \in \mathcal{K}_i(\psi) \iff \psi(\mathbf{x}) \geq i, \quad \mathbf{x} \in K^n$$

$$\mathcal{K}_k(\psi) \subseteq \mathcal{K}_{k-1}(\psi) \subseteq \cdots \subseteq \mathcal{K}_1(\psi) \subseteq \mathcal{K}_0(\psi)$$

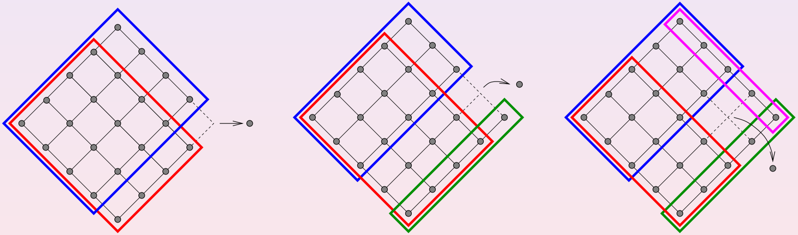
Basis

Basis of ψ at level i

$\mathbf{B}_i(\psi)$: the collection of maximal intervals in $\mathcal{K}_i(\psi)$

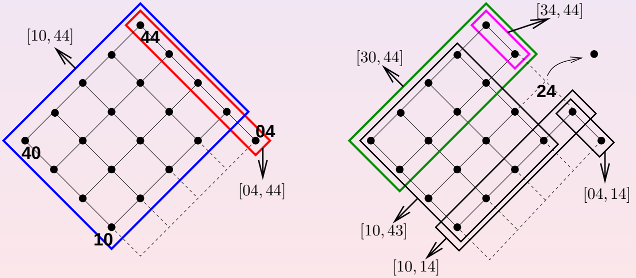
Basis computation: gray-scale operators

Number of sub-intervals resulting after removing depends on the localization of the element within the interval



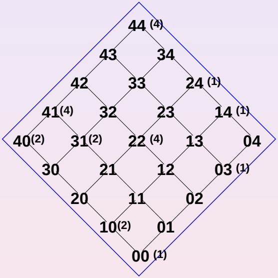
Basis computation: gray-scale operators

- 24 is both in $[10, 44]$ and $[04, 44]$. Removing 24 splits both intervals.
- Sub-interval $[34, 44]$ (resulted from splitting $[04, 44]$) is contained in sub-interval $[30, 44]$ (resulted from splitting $[10, 44]$)
- Similar effect does not happen for the binary case

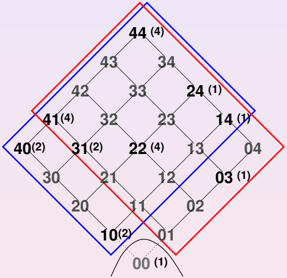


- Gray-scale case: need additional checking of redundancy

Basis computation example: gray-scale operators (1)

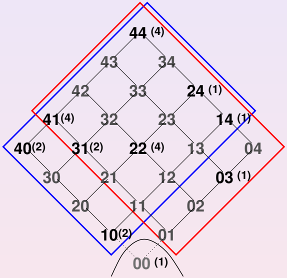


$\{[00, 44]\}$

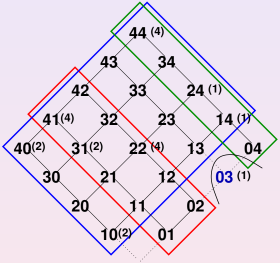


$\{[10, 44], [01, 44]\}$

Basis computation example: gray-scale operators (2)

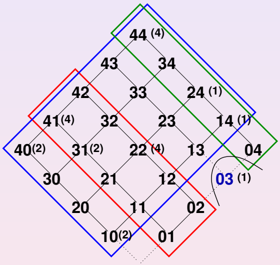


$\{[10, 44], [01, 44]\}$

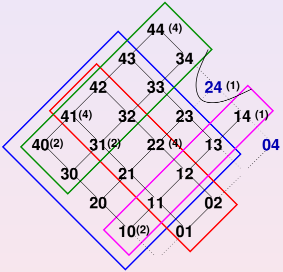


$\{[10, 44], [04, 44], [01, 42]\}$

Basis computation example: gray-scale operators (3)

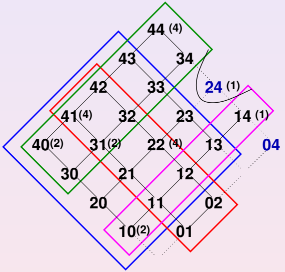


$\{[10, 44], [04, 44], [01, 42]\}$

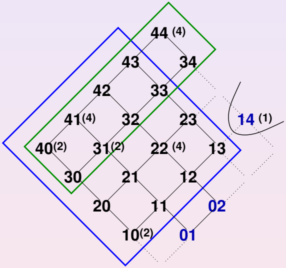


$\{[30, 44], [10, 14], [10, 43], [01, 42]\}$

Basis computation example: gray-scale operators (4)

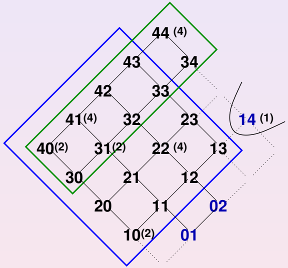


{[30, 44], [10, 14], [10, 43], [01, 42]}

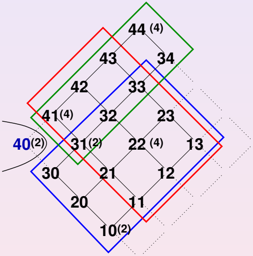


{[30, 44], [10, 43]}

Basis computation example: gray-scale operators (5)

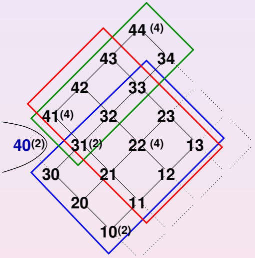


$\{[30, 44], [10, 43]\}$

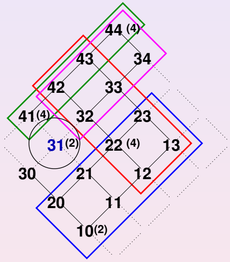


$\{[31, 44], [10, 33], [11, 43]\}$

Basis computation example: gray-scale operators (6)

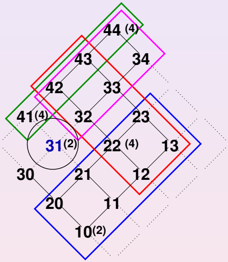


{[31, 44], [10, 33], [11, 43]}

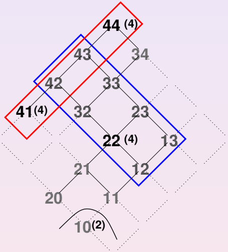


{[41, 44], [32, 44], [10, 23], [12, 43]}

Basis computation example: gray-scale operators (7)

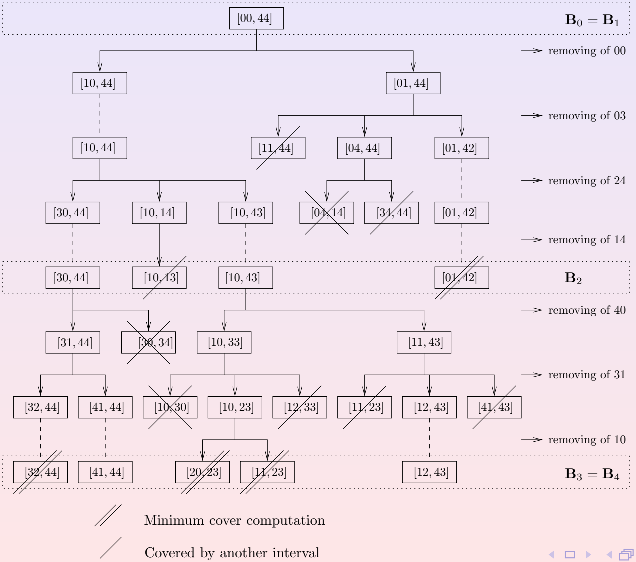


$\{[41,44],[32,44],[10,23],[12,43]\}$

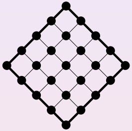


$\{[41,44],[12,43]\}$

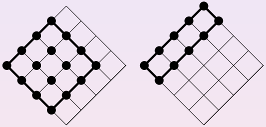
Basis computation example: gray-scale operators



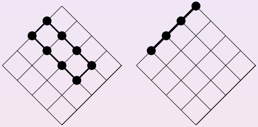
Multi-level input multi-level output operators



$\mathbf{B}_0 = \mathbf{B}_1 = \{[00, 44]\}$



$\mathbf{B}_2 = \{[10, 43], [30, 44]\}$



$\mathbf{B}_3 = \mathbf{B}_4 = \{[41, 44], [12, 43]\}$

Concluding remarks

- Given a set of elements known to be in the kernel of an operator and others known not to be in the kernel, the ISI algorithm can find minimal set of intervals that corresponds to the given elements.
- In that sense, the algorithm can be used to "learn" image operators from examples.
- This framework is more interesting to impose algebraic restrictions.

Concluding remarks

- Given a set of elements known to be in the kernel of an operator and others known not to be in the kernel, the ISI algorithm can find minimal set of intervals that corresponds to the given elements.
- In that sense, the algorithm can be used to "learn" image operators from examples.
- This framework is more interesting to impose algebraic restrictions.

Concluding remarks

- Given a set of elements known to be in the kernel of an operator and others known not to be in the kernel, the ISI algorithm can find minimal set of intervals that corresponds to the given elements.
- In that sense, the algorithm can be used to “learn” image operators from examples.
- This framework is more interesting to impose algebraic restrictions.

Concluding remarks

- Given a set of elements known to be in the kernel of an operator and others known not to be in the kernel, the ISI algorithm can find minimal set of intervals that corresponds to the given elements.
- In that sense, the algorithm can be used to "learn" image operators from examples.
- This framework is more interesting to impose algebraic restrictions.