Program Analysis (CO470/97128/97146)
Probabilistic Abstract Interpretation

Herbert Wiklicky

Department of Computing
Imperial College London

herbert@doc.ic.ac.uk
h.wiklicky@imperial.ac.uk

Autumn 2020
Galois Connections

Definition
Let $\mathcal{C} = (\mathcal{C}, \leq_\mathcal{C})$ and $\mathcal{D} = (\mathcal{D}, \leq_\mathcal{D})$ be two partially ordered sets with two order-preserving functions $\alpha : \mathcal{C} \mapsto \mathcal{D}$ and $\gamma : \mathcal{D} \mapsto \mathcal{C}$. Then $(\mathcal{C}, \alpha, \gamma, \mathcal{D})$ form a Galois connection iff

(i) $\alpha \circ \gamma$ is reductive i.e. $\forall d \in \mathcal{D}, \alpha \circ \gamma(d) \leq_\mathcal{D} d$,

(ii) $\gamma \circ \alpha$ is extensive i.e. $\forall c \in \mathcal{C}, c \leq_\mathcal{C} \gamma \circ \alpha(c)$.
Galois Connections

Definition
Let \( C = (C, \leq_C) \) and \( D = (D, \leq_D) \) be two partially ordered sets with two order-preserving functions \( \alpha : C \rightarrow D \) and \( \gamma : D \rightarrow C \). Then \((C, \alpha, \gamma, D)\) form a Galois connection iff

(i) \( \alpha \circ \gamma \) is reductive i.e. \( \forall d \in D, \alpha \circ \gamma(d) \leq_D d \),

(ii) \( \gamma \circ \alpha \) is extensive i.e. \( \forall c \in C, c \leq_C \gamma \circ \alpha(c) \).

Proposition
Let \((C, \alpha, \gamma, D)\) be a Galois connection. Then \( \alpha \) and \( \gamma \) are quasi-inverse, i.e.

(i) \( \alpha \circ \gamma \circ \alpha = \alpha \) and (ii) \( \gamma \circ \alpha \circ \gamma = \gamma \)
General Construction

The general construction of correct (and optimal) abstractions $f\#$ of concrete function $f$ is as follows:
General Construction

The general construction of correct (and optimal) abstractions $f\#$ of concrete function $f$ is as follows:

```
A ←α→ A#
|     |     |
f ↓ γ ↓ f#

B ←α'→ B#
|     |     |
γ' ↓     
```

Correct approximation: $\alpha' \circ f \leq f\# \circ \alpha$.

Induced semantics: $f\# = \alpha' \circ f \circ \gamma$.
General Construction

The general construction of correct (and optimal) abstractions $f#$ of concrete function $f$ is as follows:

\[ \begin{array}{ccc}
A & \xrightarrow{\alpha} & A^#
\\
\downarrow{f} & & \downarrow{f^#}
\\
B & \xrightarrow{\alpha'} & B^#
\end{array} \]

Correct approximation:

\[ \alpha' \circ f \leq # f^# \circ \alpha. \]
The general construction of correct (and optimal) abstractions $f\#$ of concrete function $f$ is as follows:

![Diagram of general construction]

**Correct approximation:**

$$\alpha' \circ f \leq \# f\# \circ \alpha.$$ 

**Induced semantics:**

$$f\# = \alpha' \circ f \circ \gamma.$$
A probabilistic domain is essentially a vector space which represents the distributions $\text{Dist}(\text{State}) \subseteq \mathcal{V}(\text{State})$ on the state space $\text{State}$ of a probabilistic transition system, i.e. for finite state spaces.
A probabilistic domain is essentially a vector space which represents the distributions \( \text{Dist}(\text{State}) \subseteq \mathcal{V}(\text{State}) \) on the state space \( \text{State} \) of a probabilistic transition system, i.e. for finite state spaces

\[
\mathcal{V}(\text{State}) = \{ (v_s)_{s \in \text{State}} \mid v_s \in \mathbb{R} \}.
\]
Probabilistic Abstraction Domains

A probabilistic domain is essentially a vector space which represents the distributions $\text{Dist}(\text{State}) \subseteq \mathcal{V}(\text{State})$ on the state space $\text{State}$ of a probabilistic transition system, i.e. for finite state spaces

$$\mathcal{V}(\text{State}) = \{ (v_s)_{s \in \text{State}} \mid v_s \in \mathbb{R} \}.$$ 

In the infinite setting we can identify $\mathcal{V}(\text{State})$ with the Hilbert space $\ell^2(\text{State}).$
A probabilistic domain is essentially a vector space which represents the distributions $\text{Dist} (\text{State}) \subseteq \mathcal{N} (\text{State})$ on the state space $\text{State}$ of a probabilistic transition system, i.e. for finite state spaces

$$\mathcal{N} (\text{State}) = \{ (v_s)_{s \in \text{State}} \mid v_s \in \mathbb{R} \}.$$ 

In the infinite setting we can identify $\mathcal{N} (\text{State})$ with the Hilbert space $\ell^2 (\text{State})$.

The notion of norm (distance) is essential for our treatment; we will consider normed vector spaces.
A norm on a vector space $\mathcal{V}$ is a map $\| \cdot \| : \mathcal{V} \to \mathbb{R}$ such that for all $v, w \in \mathcal{V}$ and $c \in \mathbb{C}$:

- $\|v\| \geq 0$,
- $\|v\| = 0 \iff v = o$,
- $\|cv\| = |c|\|v\|$,
- $\|v + w\| \leq \|v\| + \|w\|$.

Note: The structural similarities between distances and partial orders can be made precise (cf. Category Theory).
Norm and Distance

A norm on a vector space $\mathcal{V}$ is a map $\| \cdot \| : \mathcal{V} \to \mathbb{R}$ such that for all $v, w \in \mathcal{V}$ and $c \in \mathbb{C}$:

- $\|v\| \geq 0$,
A norm on a vector space \( \mathcal{V} \) is a map \( \| . \| : \mathcal{V} \to \mathbb{R} \) such that for all \( v, w \in \mathcal{V} \) and \( c \in \mathbb{C} \):

- \( \| v \| \geq 0 \),
- \( \| v \| = 0 \iff v = o \),
Norm and Distance

A norm on a vector space $\mathcal{V}$ is a map $\| \cdot \| : \mathcal{V} \to \mathbb{R}$ such that for all $v, w \in \mathcal{V}$ and $c \in \mathbb{C}$:

- $\| v \| \geq 0$,
- $\| v \| = 0 \iff v = 0$,
- $\| cv \| = |c| \| v \|$, 
- with $o \in \mathcal{V}$ the zero vector.

We can always use a norm to define a metric topology on a vector space via the distance function $d(v, w) = \| v - w \|$. 

Note: The structural similarities between distances and partial orders can be made precise (cf. Category Theory).
Norm and Distance

A norm on a vector space $\mathcal{V}$ is a map $\| \cdot \| : \mathcal{V} \to \mathbb{R}$ such that for all $v, w \in \mathcal{V}$ and $c \in \mathbb{C}$:

- $\| v \| \geq 0$,
- $\| v \| = 0 \iff v = o$,
- $\| cv \| = |c| \| v \|$,  
- $\| v + w \| \leq \| v \| + \| w \|$,  

with $o \in \mathcal{V}$ the zero vector.
A norm on a vector space $V$ is a map $\| \cdot \| : V \mapsto \mathbb{R}$ such that for all $v, w \in V$ and $c \in \mathbb{C}$:

- $\| v \| \geq 0$,
- $\| v \| = 0 \iff v = o$,
- $\| cv \| = |c| \| v \|$,
- $\| v + w \| \leq \| v \| + \| w \|$,

with $o \in V$ the zero vector.
A norm on a vector space $\mathcal{V}$ is a map $\|\cdot\| : \mathcal{V} \rightarrow \mathbb{R}$ such that for all $v, w \in \mathcal{V}$ and $c \in \mathbb{C}$:

1. $\|v\| \geq 0$,
2. $\|v\| = 0 \iff v = o$,
3. $\|cv\| = |c|\|v\|$,
4. $\|v + w\| \leq \|v\| + \|w\|$,

with $o \in \mathcal{V}$ the zero vector.

We can always use a norm to define a metric topology on a vector space via the distance function $d(v, w) = \|v - w\|$. 

Norm and Distance

A norm on a vector space $\mathcal{V}$ is a map $\| \cdot \| : \mathcal{V} \to \mathbb{R}$ such that for all $v, w \in \mathcal{V}$ and $c \in \mathbb{C}$:

- $\| v \| \geq 0$,
- $\| v \| = 0 \iff v = o$,
- $\| cv \| = |c| \| v \|$, 
- $\| v + w \| \leq \| v \| + \| w \|$, 

with $o \in \mathcal{V}$ the zero vector.

We can always use a norm to define a metric topology on a vector space via the distance function $d(v, w) = \| v - w \|$.

Note: The structural similarities between distances and partial orders can be made precise (cf. Category Theory).
Moore-Penrose Generalised Inverse

Definition
Let $\mathcal{C}$ and $\mathcal{D}$ be two (finite-dimensional) vector (Hilbert) spaces and $A : \mathcal{C} \to \mathcal{D}$ a linear map. Then the linear map $A^\dagger = G : \mathcal{D} \to \mathcal{C}$ is the Moore-Penrose pseudo-inverse of $A$ iff

\begin{align*}
(i) \quad & A \circ G = P_A, \\
(ii) \quad & G \circ A = P_G,
\end{align*}

where $P_A$ and $P_G$ denote orthogonal projections onto the ranges of $A$ and $G$. 
(Orthogonal) Projections – Idempotents

On finite dimensional vector (Hilbert) spaces we have an inner product $\langle \cdot, \cdot \rangle$. This measures some kind of similarity of vectors but also allows to define a norm:

$$\|x\|_2 = \sqrt{\langle x, x \rangle}$$

It also allows us to define an adjoint via:

$$\langle A(x), y \rangle = \langle x, A^*(y) \rangle$$
On finite dimensional vector (Hilbert) spaces we have an inner product \( \langle ., . \rangle \).
This measures some kind of similarity of vectors but also allows to define a norm:
\[
\|x\|_2 = \sqrt{\langle x, x \rangle}
\]
It also allows us to define an adjoint via:
\[
\langle A(x), y \rangle = \langle x, A^*(y) \rangle
\]
(Orthogonal) Projections – Idempotents

On finite dimensional vector (Hilbert) spaces we have an inner product $\langle ., . \rangle$.
This measures some kind of similarity of vectors but also allows to define a norm:
\[
\| x \|_2 = \sqrt{\langle x, x \rangle}
\]
It also allows us to define an adjoint via:
\[
\langle A(x), y \rangle = \langle x, A^*(y) \rangle
\]

- An operator $A$ is self-adjoint if $A = A^*$. 

(Orthogonal) Projections – Idempotents

On finite dimensional vector (Hilbert) spaces we have an inner product $\langle ., . \rangle$.
This measures some kind of similarity of vectors but also allows to define a norm:

$$\|x\|_2 = \sqrt{\langle x, x \rangle}$$

It also allows us to define an adjoint via:

$$\langle A(x), y \rangle = \langle x, A^*(y) \rangle$$

- An operator $A$ is self-adjoint if $A = A^*$.
- An (orthogonal) projection is a self-adjoint $E$ with $EE = E$. 
Least Squares Solutions

Corollary

Let $\mathbf{P}$ be a orthogonal projection on a finite dimensional vector space $\mathcal{V}$. Then for any $\mathbf{x} \in \mathcal{V}$, $\mathbf{P}(\mathbf{x}) = \mathbf{x}\mathbf{P}$ is the unique closest vector in $\mathcal{V}$ to $\mathbf{x}$ wrt to the Euclidean norm $\|\cdot\|_2$. 
Least Squares Solutions

Corollary

Let $\mathbf{P}$ be a orthogonal projection on a finite dimensional vector space $\mathcal{V}$. Then for any $\mathbf{x} \in \mathcal{V}$, $\mathbf{P}(\mathbf{x}) = \mathbf{xP}$ is the unique closest vector in $\mathcal{V}$ to $\mathbf{x}$ wrt to the Euclidean norm $\| \cdot \|_2$.

Definition

Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{b} \in \mathbb{R}^m$. Then $\mathbf{u} \in \mathbb{R}^n$ is called a least squares solution to $\mathbf{Ax} = \mathbf{b}$ if

$$\| \mathbf{Au} - \mathbf{b} \| \leq \| \mathbf{Av} - \mathbf{b} \|, \text{ for all } \mathbf{v} \in \mathbb{R}^n.$$
Least Squares Solutions

Corollary
Let $P$ be an orthogonal projection on a finite dimensional vector space $V$. Then for any $x \in V$, $P(x) = xP$ is the unique closest vector in $V$ to $x$ with respect to the Euclidean norm $\| \cdot \|_2$.

Definition
Let $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. Then $u \in \mathbb{R}^n$ is called a least squares solution to $Ax = b$ if

$$\|Au - b\| \leq \|Av - b\|, \text{ for all } v \in \mathbb{R}^n.$$ 

Theorem
Let $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. Then $A^\dagger b$ is the minimal least squares solution to $Ax = b$. 

Vector Space Lifting

Free vector space construction on a set $S$:

$$\mathcal{V}(S) = \left\{ \sum x_s s \mid x_s \in \mathbb{R}, s \in S \right\}$$
Vector Space Lifting

Free vector space construction on a set $S$:

$$\mathcal{V}(S) = \left\{ \sum x_s s \mid x_s \in \mathbb{R}, s \in S \right\}$$

An obvious way to lift an extraction function to a linear map between vector spaces is to construct the free vector spaces on $\mathcal{C}$ and $\mathcal{D}$ and define:
Vector Space Lifting

Free vector space construction on a set $S$:

$$\mathcal{V}(S) = \left\{ \sum x_s s \mid x_s \in \mathbb{R}, s \in S \right\}$$

An obvious way to lift an extraction function to a linear map between vector spaces is to construct the free vector spaces on $\mathcal{C}$ and $\mathcal{D}$ and define:

**Vector Space lifting**: $\vec{\alpha} : \mathcal{V}(\mathcal{C}) \rightarrow \mathcal{V}(\mathcal{D})$

$$\vec{\alpha}(p_1 \cdot \vec{c}_1 + p_2 \cdot \vec{c}_2 + \ldots) = p_i \cdot \alpha(c_1) + p_2 \cdot \alpha(c_2) \ldots$$
Vector Space Lifting

Free vector space construction on a set $S$:

$$\mathcal{V}(S) = \left\{ \sum x_s s \mid x_s \in \mathbb{R}, s \in S \right\}$$

An obvious way to lift an extraction function to a linear map between vector spaces is to construct the free vector spaces on $\mathcal{C}$ and $\mathcal{D}$ and define:

**Vector Space lifting:** $\vec{\alpha} : \mathcal{V}(\mathcal{C}) \to \mathcal{V}(\mathcal{D})$

$$\vec{\alpha}(p_1 \cdot \vec{c}_1 + p_2 \cdot \vec{c}_2 + \ldots) = p_i \cdot \alpha(c_1) + p_2 \cdot \alpha(c_2) \ldots$$

**Support Set:** $\text{supp} : \mathcal{V}(\mathcal{C}) \to \mathcal{P}(\mathcal{C})$

$$\text{supp}(\vec{x}) = \left\{ c_i \mid \langle c_i, p_i \rangle \in \vec{x} \text{ and } p_i \neq 0 \right\}$$
Lemma

Let $\vec{\alpha}$ be a probabilistic abstraction function and let $\vec{\gamma}$ be its Moore-Penrose pseudo-inverse.

Then $\vec{\gamma} \circ \vec{\alpha}$ is extensive with respect to the inclusion on the support sets of vectors in $\mathcal{V}(\mathcal{C})$, i.e. $\forall \vec{x} \in \mathcal{V}(\mathcal{C})$,

$$\text{supp}(\vec{x}) \subseteq \text{supp}(\vec{\gamma} \circ \vec{\alpha}(\vec{x})).$$
Relation with Classical Abstractions

Lemma
Let $\widetilde{\alpha}$ be a probabilistic abstraction function and let $\widetilde{\gamma}$ be its Moore-Penrose pseudo-inverse. Then $\widetilde{\gamma} \circ \widetilde{\alpha}$ is extensive with respect to the inclusion on the support sets of vectors in $\mathcal{V}(C)$, i.e. $\forall \vec{x} \in \mathcal{V}(C)$,

$$\text{supp}(\vec{x}) \subseteq \text{supp}(\widetilde{\gamma} \circ \widetilde{\alpha}(\vec{x})).$$

Analogously we can show that $\widetilde{\alpha} \circ \widetilde{\gamma}$ is reductive. Therefore,
Relation with Classical Abstractions

Lemma
Let $\hat{\alpha}$ be a probabilistic abstraction function and let $\hat{\gamma}$ be its Moore-Penrose pseudo-inverse.

Then $\hat{\gamma} \circ \hat{\alpha}$ is extensive with respect to the inclusion on the support sets of vectors in $\mathcal{V}(\mathcal{C})$, i.e. $\forall \tilde{x} \in \mathcal{V}(\mathcal{C})$,

$$\text{supp}(\tilde{x}) \subseteq \text{supp}(\hat{\gamma} \circ \hat{\alpha}(\tilde{x})).$$

Analogously we can show that $\hat{\alpha} \circ \hat{\gamma}$ is reductive. Therefore,

Proposition
$(\hat{\alpha}, \hat{\gamma})$ form a Galois connection wrt the support sets of $\mathcal{V}(\mathcal{C})$ and $\mathcal{V}(\mathcal{D})$, ordered by inclusion.
Examples of Lifted Abstractions

Parity Abstraction operator on $\forall(\{1, \ldots, n\})$ (with $n$ even):

$$A_p = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ \vdots & \vdots \\ 0 & 1 \end{pmatrix}$$
Examples of Lifted Abstractions

Parity Abstraction operator on $\mathcal{V}(\{1, \ldots, n\})$ (with $n$ even):

$$A_p = \begin{pmatrix}
1 & 0 \\
0 & 1 \\
1 & 0 \\
0 & 1 \\
\vdots & \vdots \\
0 & 1
\end{pmatrix} \quad A_p^\dagger = \begin{pmatrix}
\frac{2}{n} & 0 & \frac{2}{n} & 0 & \cdots & 0 \\
0 & \frac{2}{n} & 0 & \frac{2}{n} & \cdots & \frac{2}{n}
\end{pmatrix}$$
Examples of Lifted Abstractions

Sign Abstraction operator on $\mathcal{V}(\{-n, \ldots, 0, \ldots, n\})$: 

\[
A_s = \begin{pmatrix} 
1 & 0 & 0 \\
\vdots & \vdots & \vdots \\
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
\vdots & \vdots & \vdots \\
0 & 0 & 1 
\end{pmatrix}
\]
Examples of Lifted Abstractions

Sign Abstraction operator on $\mathcal{V}(\{-n, \ldots, 0, \ldots, n\})$:

$$A_s = \begin{pmatrix}
1 & 0 & 0 \\
\vdots & \vdots & \vdots \\
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
\vdots & \vdots & \vdots \\
0 & 0 & 1 \\
\end{pmatrix} \quad A_s^\dagger = \begin{pmatrix}
\frac{1}{n} & \ldots & \frac{1}{n} & 0 & 0 & \ldots & 0 \\
0 & \ldots & 0 & 1 & 0 & \ldots & 0 \\
0 & \ldots & 0 & 0 & \frac{1}{n} & \ldots & \frac{1}{n} \\
\end{pmatrix}$$
Concrete and abstract domain are step-functions on \([a, b]\).
Concrete and abstract domain are step-functions on \([a, b]\). The set of (real-valued) step-function \(\mathcal{T}_n\) is based on the sub-division of the interval into \(n\) sub-intervals.
Concrete and abstract domain are step-functions on \([a, b]\). The set of (real-valued) step-function \(T_n\) is based on the sub-division of the interval into \(n\) sub-intervals.
Concrete and abstract domain are step-functions on \([a, b]\). The set of (real-valued) step-function \(\mathcal{T}_n\) is based on the sub-division of the interval into \(n\) sub-intervals.

Each step function in \(\mathcal{T}_n\) corresponds to a vector in \(\mathbb{R}^n\), e.g.
Concrete and abstract domain are step-functions on \([a, b]\). The set of (real-valued) step-function \(\mathcal{T}_n\) is based on the sub-division of the interval into \(n\) sub-intervals.

Each step function in \(\mathcal{T}_n\) corresponds to a vector in \(\mathbb{R}^n\), e.g.

\[
\begin{pmatrix}
5 & 5 & 6 & 7 & 8 & 4 & 3 & 2 & 8 & 6 & 6 & 7 & 9 & 8 & 8 & 7
\end{pmatrix}
\]
Example: Abstraction Matrices

\[ A_8 = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix} \]
Example: Abstraction Matrices

\[ G_8 = \begin{pmatrix}
\frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\
\end{pmatrix} \]
Approximation Estimates

Compute the *least square error* as

$$\| f - f_{AG} \|.$$
Approximation Estimates

Compute the *least square error* as

\[ \| f - fAG \|. \]

\[
\begin{align*}
\| f - fA_8G_8 \| &= 3.5355 \\
\| f - fA_4G_4 \| &= 5.3151 \\
\| f - fA_2G_2 \| &= 5.9896 \\
\| f - fA_1G_1 \| &= 7.6444
\end{align*}
\]
The tensor product of $n$ linear operators $A_1, A_2, \ldots, A_n$ is associative (but in general not commutative) and has e.g. the following properties:
Tensor Product Properties

The tensor product of \( n \) linear operators \( A_1, A_2, \ldots, A_n \) is associative (but in general not commutative) and has e.g. the following properties:

1. \((A_1 \otimes \ldots \otimes A_n) \cdot (B_1 \otimes \ldots \otimes B_n) = A_1 \cdot B_1 \otimes \ldots \otimes A_n \cdot B_n\)
Tensor Product Properties

The tensor product of $n$ linear operators $A_1, A_2, \ldots, A_n$ is associative (but in general not commutative) and has e.g. the following properties:

1. $(A_1 \otimes \ldots \otimes A_n) \cdot (B_1 \otimes \ldots \otimes B_n) =$
   $$= A_1 \cdot B_1 \otimes \ldots \otimes A_n \cdot B_n$$

2. $A_1 \otimes \ldots \otimes (\alpha A_i) \otimes \ldots \otimes A_n =$
   $$= \alpha (A_1 \otimes \ldots \otimes A_i \otimes \ldots \otimes A_n)$$
Tensor Product Properties

The tensor product of $n$ linear operators $A_1, A_2, \ldots, A_n$ is associative (but in general not commutative) and has e.g. the following properties:

1. $(A_1 \otimes \ldots \otimes A_n) \cdot (B_1 \otimes \ldots \otimes B_n) =$
   \[ = A_1 \cdot B_1 \otimes \ldots \otimes A_n \cdot B_n \]

2. $A_1 \otimes \ldots \otimes (\alpha A_i) \otimes \ldots \otimes A_n =$
   \[ = \alpha (A_1 \otimes \ldots \otimes A_i \otimes \ldots \otimes A_n) \]

3. $A_1 \otimes \ldots \otimes (A_i + B_i) \otimes \ldots \otimes A_n =$
   \[ = (A_1 \otimes \ldots \otimes A_i \otimes \ldots \otimes A_n) + (A_1 \otimes \ldots \otimes B_i \otimes \ldots \otimes A_n) \]
Tensor Product Properties

The tensor product of \( n \) linear operators \( A_1, A_2, \ldots, A_n \) is associative (but in general not commutative) and has e.g. the following properties:

1. \( (A_1 \otimes \ldots \otimes A_n) \cdot (B_1 \otimes \ldots \otimes B_n) = A_1 \cdot B_1 \otimes \ldots \otimes A_n \cdot B_n \)
2. \( A_1 \otimes \ldots \otimes (\alpha A_i) \otimes \ldots \otimes A_n = \alpha (A_1 \otimes \ldots \otimes A_i \otimes \ldots \otimes A_n) \)
3. \( A_1 \otimes \ldots \otimes (A_i + B_i) \otimes \ldots \otimes A_n = (A_1 \otimes \ldots \otimes A_i \otimes \ldots \otimes A_n) + (A_1 \otimes \ldots \otimes B_i \otimes \ldots \otimes A_n) \)
4. \( (A_1 \otimes \ldots \otimes A_i \otimes \ldots \otimes A_n)^\dagger = A_1^\dagger \otimes \ldots \otimes A_i^\dagger \otimes \ldots \otimes A_n^\dagger \)
Moore-Penrose Pseudo-Inverse of a Tensor Product is:

\[(A_1 \otimes A_2 \otimes \ldots \otimes A_n)^\dagger = A_1^\dagger \otimes A_2^\dagger \otimes \ldots \otimes A_n^\dagger\]
Abstract Semantics

Moore-Penrose Pseudo-Inverse of a Tensor Product is:

\[(A_1 \otimes A_2 \otimes \ldots \otimes A_n)^\dagger = A_1^\dagger \otimes A_2^\dagger \otimes \ldots \otimes A_n^\dagger\]

Via linearity we can construct \(T^\#\) in the same way as \(T\), i.e

\[T^\#(P) = \sum_{\langle i, p_{ij}, j \rangle \in \mathcal{F}(P)} p_{ij} \cdot T^\#(\ell_i, \ell_j)\]

with local abstraction of individual variables:

\[T^\#(\ell_i, \ell_j) = (A_1^\dagger N_{i1} A_1) \otimes (A_2^\dagger N_{i2} A_2) \otimes \ldots \otimes (A_v^\dagger N_{iv} A_v) \otimes M_{ij}\]
Argument

\[ T^\# = A^\dagger T A \]
\[ T^\# = A^\dagger TA = A^\dagger \left( \sum_{i,j} T(i,j) \right) A \]
\[ T^\# = A^\dagger TA = A^\dagger \left( \sum_{i,j} T(i,j) \right) A = \sum_{i,j} A^\dagger T(i,j) A \]
Argument

\[
T# = A^\dagger TA \\
= A^\dagger (\sum_{i,j} T(i,j))A \\
= \sum_{i,j} A^\dagger T(i,j)A \\
= \sum_{i,j} (\bigotimes_{k} A_k)^\dagger T(i,j)(\bigotimes_{k} A_k)
\]
\[ T^\# = A^\dagger T A \]
\[ = A^\dagger \left( \sum_{i,j} T(i,j) \right) A \]
\[ = \sum_{i,j} A^\dagger T(i,j) A \]
\[ = \sum_{i,j} \left( \bigotimes_k A_k \right)^\dagger T(i,j) \left( \bigotimes_k A_k \right) \]
\[ = \sum_{i,j} \left( \bigotimes_k A_k \right)^\dagger \left( \bigotimes_k N_{ik} \right) \left( \bigotimes_k A_k \right) \]
\[
\begin{align*}
T^\# &= \mathbf{A}^\dagger \mathbf{T} \mathbf{A} \\
&= \mathbf{A}^\dagger \left( \sum_{i,j} T(i,j) \right) \mathbf{A} \\
&= \sum_{i,j} \mathbf{A}^\dagger T(i,j) \mathbf{A} \\
&= \sum_{i,j} \left( \bigotimes_k A_k \right)^\dagger T(i,j) \left( \bigotimes_k A_k \right) \\
&= \sum_{i,j} \left( \bigotimes_k A_k \right)^\dagger \left( \bigotimes_k N_{ik} \right) \left( \bigotimes_k A_k \right) \\
&= \sum_{i,j} \bigotimes_k \left( A_k^\dagger N_{ik} A_k \right)
\end{align*}
\]
Parity Analysis

Determine at each program point whether a variable is *even* or *odd*. 
Determine at each program point whether a variable is *even* or *odd*.

**Parity Abstraction** operator on $\mathcal{V}(\{0, \ldots, n\})$ (with $n$ even):

$$A_p = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ \vdots & \vdots \\ 0 & 1 \end{pmatrix} \quad \quad A^\dagger = \begin{pmatrix} \frac{2}{n} & 0 & \frac{2}{n} & 0 & \cdots & 0 \\ 0 & \frac{2}{n} & 0 & \frac{2}{n} & \cdots & \frac{2}{n} \end{pmatrix}$$
Example

1: \( [m \leftarrow i]^1 \);
2: \textbf{while} \( [n > 1]^2 \) \textbf{do}
3: \( [m \leftarrow m \times n]^3 \);
4: \( [n \leftarrow n - 1]^4 \)
5: \textbf{end while}
6: \[ \text{stop}^5 \]
Example

1: \[m \leftarrow i\]^1;
2: while \[n > 1\]^2 do
3: \[m \leftarrow m \times n\]^3;
4: \[n \leftarrow n - 1\]^4
5: end while
6: [stop]^5

\[
T = U(m \leftarrow i) \otimes E(1, 2) + P(n > 1) \otimes E(2, 3) + P(n \leq 1) \otimes E(2, 5) + U(m \leftarrow m \times n) \otimes E(3, 4) + U(n \leftarrow n - 1) \otimes E(4, 2) + I \otimes E(5, 5)
\]
Example

1: $[m ← i]^1$;
2: while $[n > 1]^2$ do
3: $[m ← m × n]^3$;
4: $[n ← n − 1]^4$
5: end while
6: [stop]$^5$

\[ T# = U#(m ← i) ⊗ E(1, 2) + P#(n > 1) ⊗ E(2, 3) + P#(n ≤ 1) ⊗ E(2, 5) + U#(m ← m × n) ⊗ E(3, 4) + U#(n ← n − 1) ⊗ E(4, 2) + I# ⊗ E(5, 5) \]
Abstract Semantics

Abstraction: \( A = A_p \otimes I \), i.e. \( m \) abstract (parity) but \( n \) concrete.

\[
\begin{align*}
T^\# & = U^\# (m \leftarrow 1) \otimes E(1, 2) \\
& + P^\# (n > 1) \otimes E(2, 3) \\
& + P^\# (n \leq 1) \otimes E(2, 5) \\
& + U^\# (m \leftarrow m \times n) \otimes E(3, 4) \\
& + U^\# (n \leftarrow n - 1) \otimes E(4, 2) \\
& + I^\# \otimes E(5, 5)
\end{align*}
\]
Abstract Semantics

\[ U^\#(m \leftarrow 1) = \]

\[ = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 & 0 & 0 & \ldots & 0 \\ 0 & 1 & 0 & 0 & \ldots & 0 \\ 0 & 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \ldots & 1 \end{pmatrix} \]
Abstract Semantics

\[ \mathbf{U}^\#(n \leftarrow n - 1) = \]  
\[ = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 0 & 0 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 0 \end{pmatrix} \]
Abstract Semantics

$$P^\#(n > 1) =$$

$$= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$
Abstract Semantics

\( P^\#(n \leq 1) = \)

\[
\begin{pmatrix}
1 & 0 \\
0 & 1
\end{pmatrix}
\otimes
\begin{pmatrix}
1 & 0 & 0 & 0 & \ldots & 0 \\
0 & 1 & 0 & 0 & \ldots & 0 \\
0 & 0 & 0 & 0 & \ldots & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & 0 & \ldots & 0
\end{pmatrix}
\]
Abstract Semantics

\[ U^\#(m \leftarrow m \times n) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 & \ldots & 0 \\ 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & 1 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \otimes \begin{pmatrix} 0 & 0 & \ldots & 0 \\ 0 & 0 & \ldots & 0 \\ 0 & 1 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & 1 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 0 & 0 & \ldots & 0 \\ 0 & 0 & \ldots & 0 \\ 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & 1 \end{pmatrix} \]
Implementation

Implementation of concrete and abstract semantics of Factorial using octave. Ranges: $n \in \{1, \ldots, d\}$ and $m \in \{1, \ldots, d!\}$. 
Implementation of concrete and abstract semantics of Factorial using octave. Ranges: $n \in \{1, \ldots, d\}$ and $m \in \{1, \ldots, d!\}$.

<table>
<thead>
<tr>
<th>$d$</th>
<th>dim($T(F)$)</th>
<th>dim($T^#(F)$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>45</td>
<td>30</td>
</tr>
<tr>
<td>3</td>
<td>140</td>
<td>40</td>
</tr>
<tr>
<td>4</td>
<td>625</td>
<td>50</td>
</tr>
<tr>
<td>5</td>
<td>3630</td>
<td>60</td>
</tr>
<tr>
<td>6</td>
<td>25235</td>
<td>70</td>
</tr>
<tr>
<td>7</td>
<td>201640</td>
<td>80</td>
</tr>
<tr>
<td>8</td>
<td>1814445</td>
<td>90</td>
</tr>
<tr>
<td>9</td>
<td>18144050</td>
<td>100</td>
</tr>
</tbody>
</table>

Using uniform initial distributions $d_0$ for $n$ and $m$. 
Scalablity

The abstract probabilities for $m$ being even or odd when we execute the abstract program for various $d$ values are:

<table>
<thead>
<tr>
<th>$d$</th>
<th>even</th>
<th>odd</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.81818</td>
<td>0.18182</td>
</tr>
<tr>
<td>100</td>
<td>0.98019</td>
<td>0.019802</td>
</tr>
<tr>
<td>1000</td>
<td>0.99800</td>
<td>0.0019980</td>
</tr>
<tr>
<td>10000</td>
<td>0.99980</td>
<td>0.00019998</td>
</tr>
</tbody>
</table>
Define a partial order on self-adjoint operators and projections as follows: $H \sqsubseteq K$ iff $K - H$ is positive, i.e. there exists a $B$ such that $K - H = B^*B$. 

Alternatively, order projections by inclusion of their image spaces, i.e. $E \sqsubseteq F$ iff $Y_E \subseteq Y_F$.
Define a \textbf{partial order} on self-adjoint operators and projections as follows: \( H \sqsubseteq K \) iff \( K - H \) is positive, i.e. there exists a \( B \) such that \( K - H = B^*B \).

Alternatively, order projections by inclusion of their image spaces, i.e. \( E \sqsubseteq F \) iff \( Y_E \subseteq Y_F \).
Define a partial order on self-adjoint operators and projections as follows: $H \sqsubseteq K$ iff $K - H$ is positive, i.e. there exists a $B$ such that $K - H = B^*B$.

Alternatively, order projections by inclusion of their image spaces, i.e. $E \sqsubseteq F$ iff $Y_E \subseteq Y_F$.

The orthogonal projections form a complete (ortho)lattice.

The range of the intersection $E \cap F$ is to the closure of the intersection of the image spaces of $E$ and $F$.

The union $E \sqcup F$ corresponds to the union of the images.
Associate to every Probabilistic Abstract Interpretation \((A, G)\) a projection, similar to so-called “upper closure operators” (uco):

\[ E = AG = AA^\dagger. \]
Computing Intersections/Unions

Associate to every Probabilistic Abstract Interpretation \((A, G)\) a projection, similar to so-called “upper closure operators” (uco):

\[ E = AG = AA^\dagger. \]

A general way to construct \(E \cap F\) and (by exploiting de Morgan’s law) also \(E \cup F = (E^\perp \cap F^\perp)^\perp\) is via an infinite approximation sequence and has been suggested by Halmos:

\[ E \cap F = \lim_{n \to \infty} (EFE)^n. \]
Commutative Case

The concrete construction of $E \sqcup F$ and $E \sqcap F$ is in general not trivial. Only for commuting projections we have:

$$E \sqcup F = E + F - EF \text{ and } E \sqcap F = EF.$$
The concrete construction of $E \boxplus F$ and $E \sqcap F$ is in general not trivial. Only for commuting projections we have:

$$E \boxplus F = E + F - EF \text{ and } E \sqcap F = EF.$$ 

**Example**

Consider a finite set $\Omega$ with a probability structure. For any (measurable) subset $A$ of $\Omega$ define the characteristic function $\chi_A$ with $\chi_A(x) = 1$ if $x \in A$ and 0 otherwise.
Commutative Case

The concrete construction of $E \sqcup F$ and $E \sqcap F$ is in general not trivial. Only for commuting projections we have:

$$E \sqcup F = E + F - EF \quad \text{and} \quad E \sqcap F = EF.$$ 

Example

Consider a finite set $\Omega$ with a probability structure. For any (measurable) subset $A$ of $\Omega$ define the characteristic function $\chi_A$ with $\chi_A(x) = 1$ if $x \in A$ and 0 otherwise. The characteristic functions are (commutative) projections on random variables using pointwise multiplication, i.e. $X\chi_A\chi_A = X\chi_A$. We have $\chi_{A \cap B} = \chi_A\chi_B$ and $\chi_{A \cup B} = \chi_A + \chi_B - \chi_A\chi_B.$
The Moore-Penrose pseudo-inverse is also useful for computing the $E \sqcap F$ and $E \sqcup F$ of general, non-commuting projections via the parallel sum

$$A : B = A(A + B)^\dagger B$$

The intersection of projections is given by:

$$E \sqcap F = 2(E : F) = E(E + F)^\dagger F + F(E + F)^\dagger E$$

Consider a "duel" between two cowboys:

- Cowboy A – hitting probability $a$
- Cowboy B – hitting probability $b$
Consider a "duel" between two cowboys:

- Cowboy $A$ – hitting probability $a$
- Cowboy $B$ – hitting probability $b$

1. Choose (non-deterministically) whether $A$ or $B$ starts.
Consider a "duel" between two cowboys:

- Cowboy $A$ – hitting probability $a$
- Cowboy $B$ – hitting probability $b$

1. Choose (non-deterministically) whether $A$ or $B$ starts.
2. Repeat until winner is known:
Variable Probabilities: Duel at High Noon

Consider a "duel" between two cowboys:

- Cowboy A – hitting probability \( a \)
- Cowboy B – hitting probability \( b \)

1. Choose (non-deterministically) whether A or B starts.
2. Repeat until winner is known:
   - If it is A’s turn he will hit/shoot B with probability \( a \);
     If B is shot then A is the winner, otherwise it’s B’s turn.

Question: What is the life expectancy of A or B?

Question: What happens if A is learning to shoot better during the duel? How can we model dynamic probabilities?

Introduced by McIver and Morgan (2005).
Discussed in detail by Gretz, Katoen, McIver (2012/14)
Variable Probabilities: Duel at High Noon

Consider a "duel" between two cowboys:

- Cowboy $A$ – hitting probability $a$
- Cowboy $B$ – hitting probability $b$

1. Choose (non-deterministically) whether $A$ or $B$ starts.
2. Repeat until winner is known:
   - If it is $A$’s turn he will hit/shoot $B$ with probability $a$;
     If $B$ is shot then $A$ is the winner, otherwise it’s $B$’s turn.
   - If it is $B$’s turn he will hit/shoot $A$ with probability $b$;
     If $A$ is shot then $B$ is the winner, otherwise it’s $A$’s turn.

Question: What is the life expectancy of $A$ or $B$?

Question: What happens if $A$ is learning to shoot better during the duel? How can we model dynamic probabilities?

Introduced by McIver and Morgan (2005).
Discussed in detail by Gretz, Katoen, McIver (2012/14)
Consider a "duel" between two cowboys:

- Cowboy A – hitting probability $a$
- Cowboy B – hitting probability $b$

1. Choose (non-deterministically) whether A or B starts.
2. Repeat until winner is known:
   - If it is A’s turn he will hit/shoot B with probability $a$;
     If B is shot then A is the winner, otherwise it’s B’s turn.
   - If it is B’s turn he will hit/shoot A with probability $b$;
     If A is shot then B is the winner, otherwise it’s A’s turn.

Question: What is the life expectancy of A or B?

Introduced by McIver and Morgan (2005).
Discussed in detail by Gretz, Katoen, McIver (2012/14)
Consider a "duel" between two cowboys:

- Cowboy \( A \) – hitting probability \( a \)
- Cowboy \( B \) – hitting probability \( b \)

1. Choose (non-deterministically) whether \( A \) or \( B \) starts.
2. Repeat until winner is known:
   - If it is \( A \)'s turn he will hit/shoot \( B \) with probability \( a \);
     If \( B \) is shot then \( A \) is the winner, otherwise it’s \( B \)’s turn.
   - If it is \( B \)’s turn he will hit/shoot \( A \) with probability \( b \);
     If \( A \) is shot then \( B \) is the winner, otherwise it’s \( A \)’s turn.

**Question:** What is the life expectancy of \( A \) or \( B \)?

**Question:** What happens if \( A \) is learning to shoot better during the duel? How can we model dynamic probabilities?

Introduced by McIver and Morgan (2005).
Discussed in detail by Gretz, Katoen, McIver (2012/14)
Example: Duelling Cowboys

begin
# who’s first turn
choose 1:{t:=0} or 1:{t:=1} ro;
# continue until ...
c := 1;
while c == 1 do
  if (t==0) then
    choose ak:{c:=0} or am:{t:=1} ro
  else
    choose bk:{c:=0} or bm:{t:=0} ro
  fi;
end;
stop; # terminal loop
Example: Duelling Cowboys

The survival chances, i.e. winning probability, for A.
References

References


References


References


References


Herbert Wiklicky: *On Dynamical Probabilities, or: How to learn to shoot straight*. Coordinations, LNCS 9686, 2016.