Formal methods for accountable decision making

Nicola Botta[1] (with P. Jansson[2] and C. Ionescu[3])

[1]Potsdam Institute for Climate Impact Research
[2]Chalmers University of Technology, Gothenburg, Sweden
[3]University of Oxford, Oxford, UK

Contact

- Mail: botta@pik-potsdam.de
- Phone: 288-2657
- Office: A56, 3.26 (Sonne)

- These slides: https://gitlab.pik-potsdam.de/botta/IdrisLibs/tree/master/lectures/2018-12-03.PIK.Global_change_management_week

Outline

- Decision problems in climate research
- A conceptual emission problem
- An environment for specifying and solving SDPs
- Specifying and solving the emission problem

The context

The context

- Global GHG emissions have to be reduced fast to avoid unmanageable impacts of climate change.

The context

- Global GHG emissions have to be reduced fast to avoid unmanageable impacts of climate change.
- Too rapid reductions may compromise the wealth of one or more upcoming generations but ...

The context

- Global GHG emissions have to be reduced fast to avoid unmanageable impacts of climate change.
- Too rapid reductions may compromise the wealth of one or more upcoming generations but ...
- ... they may promote a transition to societies that are more wealthy, safe, fair and manageable.

The context

- Global GHG emissions have to be reduced fast to avoid unmanageable impacts of climate change.
- Too rapid reductions may compromise the wealth of one or more upcoming generations but ...
- ... they may promote a transition to societies that are more wealthy, safe, fair and manageable.
- New technologies that significantly reduce the costs of very fast emission reductions may become available soon.

The context

- Emission reductions imply different costs and benefits for different countries.

The context

- Emission reductions imply different costs and benefits for different countries.

- The highest global benefits are obtained if most countries reduce emissions by certain (optimal, fair, ...) country-specific amounts.

The context

- Emission reductions imply different costs and benefits for different countries.
- The highest global benefits are obtained if most countries reduce emissions by certain (optimal, fair, ...) country-specific amounts.
- In this situation most countries have a free-ride opportunity!

Decision problems and dilemmas

Decision problems and dilemmas

- Decision problems often have the form of a dilemma.

Decision problems and dilemmas

- Decision problems often have the form of a dilemma.
- The consequences of decisions are typically uncertain.

Decision problems and dilemmas

- Decision problems often have the form of a dilemma.
- The consequences of decisions are typically uncertain.
- Often, decisions are taken (and possibly revised) sequentially as time unfolds.

Decision problems and dilemmas

- Decision problems often have the form of a dilemma.
- The consequences of decisions are typically uncertain.
- Often, decisions are taken (and possibly revised) sequentially as time unfolds.
- Often, decisions are implemented with delays or they are not implemented at all.

Decision problems and dilemmas

- Decision problems often have the form of a dilemma.
- The consequences of decisions are typically uncertain.
- Often, decisions are taken (and possibly revised) sequentially as time unfolds.
- Often, decisions are implemented with delays or they are not implemented at all.
- Typically, policy advice entails two problems: what are "best" decisions and . . . .

Decision problems and dilemmas

- Decision problems often have the form of a dilemma.
- The consequences of decisions are typically uncertain.
- Often, decisions are taken (and possibly revised) sequentially as time unfolds.
- Often, decisions are implemented with delays or they are not implemented at all.
- Typically, policy advice entails two problems: what are "best" decisions and . . . how to implement them.

A conceptual emission problem

A conceptual emission problem

- A decision maker has to take a sequence of decisions about GHG emissions, one after the other.

A conceptual emission problem

- A decision maker has to take a sequence of decisions about GHG emissions, one after the other.
- At each decision step, it can only pick up one of two options: low emissions or high emissions.

A conceptual emission problem

- A decision maker has to take a sequence of decisions about GHG emissions, one after the other.
- At each decision step, it can only pick up one of two options: low emissions or high emissions.
- The decision taken by the decision maker may or may not be implemented during the time until the next decision has to be taken.

A conceptual emission problem

- A decision maker has to take a sequence of decisions about GHG emissions, one after the other.
- At each decision step, it can only pick up one of two options: low emissions or high emissions.
- The decision taken by the decision maker may or may not be implemented during the time until the next decision has to be taken.
- If implemented, low emissions increase cumulated emissions less than high emissions.

A conceptual emission problem

- At each step, the decision maker has to choose between low and high emissions on the basis of four data:

A conceptual emission problem

- At each step, the decision maker has to choose between low and high emissions on the basis of four data:
  - The amount of cumulated emissions.

A conceptual emission problem

- At each step, the decision maker has to choose between low and high emissions on the basis of four data:
  - The amount of cumulated emissions.
  - The current emission level.

A conceptual emission problem

- At each step, the decision maker has to choose between low and high emissions on the basis of four data:
  - The amount of cumulated emissions.
  - The current emission level.
  - The availability of technologies for reducing emissions.

A conceptual emission problem

- At each step, the decision maker has to choose between low and high emissions on the basis of four data:
  - The amount of cumulated emissions.
  - The current emission level.
  - The availability of technologies for reducing emissions.
  - A state of the world which can be either good or bad.

A conceptual emission problem

- At the first decision step, the decision maker observes zero cumulated emissions, high current emissions, unavailable technologies and a good world.

A conceptual emission problem

- At the first decision step, the decision maker observes zero cumulated emissions, high current emissions, unavailable technologies and a good world.
- In this state, the probability that the world turns bad is low.

A conceptual emission problem

- At the first decision step, the decision maker observes zero cumulated emissions, high current emissions, unavailable technologies and a good world.
- In this state, the probability that the world turns bad is low.
- But if the cumulated emissions increase beyond a critical threshold, the probability that the world becomes bad steeply increases.

A conceptual emission problem

- At the first decision step, the decision maker observes zero cumulated emissions, high current emissions, unavailable technologies and a good world.
- In this state, the probability that the world turns bad is low.
- But if the cumulated emissions increase beyond a critical threshold, the probability that the world becomes bad steeply increases.
- Once the world has reached a bad state, there is no chance to turn back to a good state.

A conceptual emission problem

- At the first decision step, the decision maker observes zero cumulated emissions, high current emissions, unavailable technologies and a good world.
- In this state, the probability that the world turns bad is low.
- But if the cumulated emissions increase beyond a critical threshold, the probability that the world becomes bad steeply increases.
- Once the world has reached a bad state, there is no chance to turn back to a good state.
- Similarly, the probability that new technologies become available is low at the first decision step. It increases steeply after a critical number of steps.

A conceptual emission problem

- At the first decision step, the decision maker observes zero cumulated emissions, high current emissions, unavailable technologies and a good world.
- In this state, the probability that the world turns bad is low.
- But if the cumulated emissions increase beyond a critical threshold, the probability that the world becomes bad steeply increases.
- Once the world has reached a bad state, there is no chance to turn back to a good state.
- Similarly, the probability that new technologies become available is low at the first decision step. It increases steeply after a critical number of steps.
- Once available, technologies stay available for ever.

A conceptual emission problem

- Being in a bad world yields less benefits than being in a good world.

A conceptual emission problem

- Being in a bad world yields less benefits than being in a good world.
- Low current emissions yield less benefits (more costs, less growth) than high current emissions.

A conceptual emission problem

- Being in a bad world yields less benefits than being in a good world.
- Low current emissions yield less benefits (more costs, less growth) than high current emissions.
- Implementing low emissions when technologies are unavailable costs more than implementing emissions when technologies are available.

A conceptual emission problem

- Being in a bad world yields less benefits than being in a good world.
- Low current emissions yield less benefits (more costs, less growth) than high current emissions.
- Implementing low emissions when technologies are unavailable costs more than implementing emissions when technologies are available.
- The decision maker aim at maximising a sum of the benefits over all decision steps.

A conceptual emission problem: wrap-up

A conceptual emission problem: wrap-up

- We have seen a simple example of a *Sequential Decision Problem* (SDP).

A conceptual emission problem: wrap-up

- We have seen a simple example of a *Sequential Decision Problem* (SDP).
- The problem has been described through informal narratives.

A conceptual emission problem: wrap-up

- We have seen a simple example of a *Sequential Decision Problem* (SDP).
- The problem has been described through informal narratives.
- Even if we make these narrative more precise, we are far from having a good understanding of the problem.

A conceptual emission problem: wrap-up

- We have seen a simple example of a *Sequential Decision Problem* (SDP).
- The problem has been described through informal narratives.
- Even if we make these narrative more precise, we are far from having a good understanding of the problem.
- Solving the problem and advising the decision maker requires answering a few obvious questions:

A conceptual emission problem: wrap-up

- We have seen a simple example of a *Sequential Decision Problem* (SDP).
- The problem has been described through informal narratives.
- Even if we make these narrative more precise, we are far from having a good understanding of the problem.
- Solving the problem and advising the decision maker requires answering a few obvious questions:
  - What kind of solutions or advice can we offer to the decision maker?

A conceptual emission problem: wrap-up

- We have seen a simple example of a *Sequential Decision Problem* (SDP).
- The problem has been described through informal narratives.
- Even if we make these narrative more precise, we are far from having a good understanding of the problem.
- Solving the problem and advising the decision maker requires answering a few obvious questions:
  - What kind of solutions or advice can we offer to the decision maker?
  - What kind of guarantees can we provide for such solutions? Can we check that they are correct? What does this mean?

A conceptual emission problem: wrap-up

- In order to answer these questions, it is useful to abstract away from the specific problem!

A conceptual emission problem: wrap-up

- In order to answer these questions, it is useful to abstract away from the specific problem!
- We start with a visual sketch of SDPs in general.

A conceptual emission problem: wrap-up

- In order to answer these questions, it is useful to abstract away from the specific problem!
- We start with a visual sketch of SDPs in general.
- Then we discuss how to specify and solve SDPs.

A conceptual emission problem: wrap-up

- In order to answer these questions, it is useful to abstract away from the specific problem!
- We start with a visual sketch of SDPs in general.
- Then we discuss how to specify and solve SDPs.
- We specify and solve the emission problem and discuss two rigorous results:

A conceptual emission problem: wrap-up

- In order to answer these questions, it is useful to abstract away from the specific problem!
- We start with a visual sketch of SDPs in general.
- Then we discuss how to specify and solve SDPs.
- We specify and solve the emission problem and discuss two rigorous results:
  - More uncertainty about the implementability of decisions dictates earlier emission reductions.

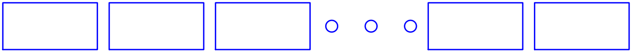
A conceptual emission problem: wrap-up

- In order to answer these questions, it is useful to abstract away from the specific problem!
- We start with a visual sketch of SDPs in general.
- Then we discuss how to specify and solve SDPs.
- We specify and solve the emission problem and discuss two rigorous results:
  - More uncertainty about the implementability of decisions dictates earlier emission reductions.
  - More uncertainty about the implications of exceeding critical thresholds make earlier reductions sub-optimal.

Sequential decision problems: a visual sketch

Sequential decision problems: a visual sketch

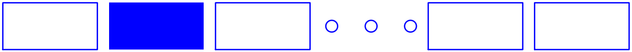
There are $n + 1$ decision steps to go ...



n+1 steps left

Sequential decision problems: a visual sketch

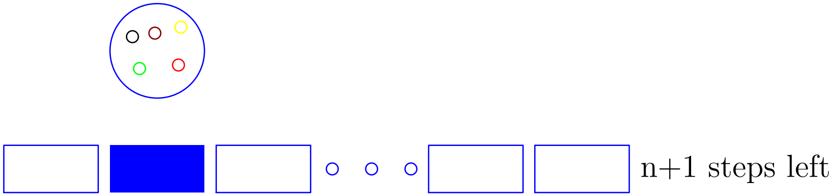
…here is the current state,



n+1 steps left

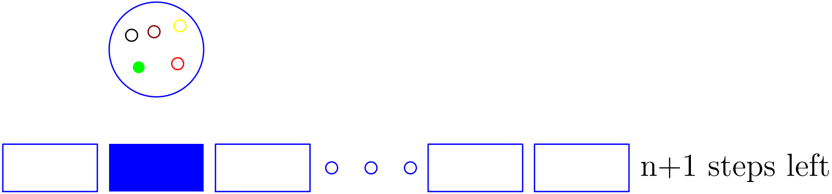
Sequential decision problems: a visual sketch

. . . here are your options.



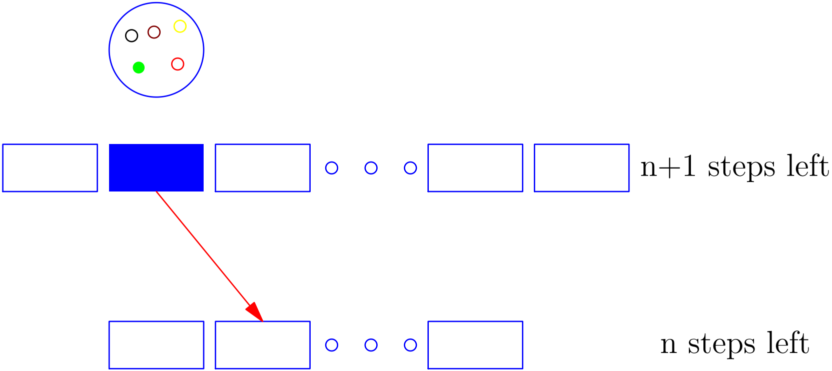
Sequential decision problems: a visual sketch

Pick one!



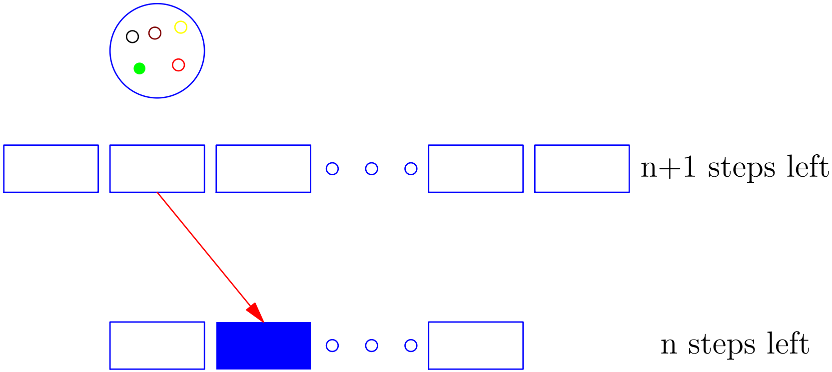
Sequential decision problems: a visual sketch

Move to a new state and . . .



Sequential decision problems: a visual sketch

. . . collect rewards and face the next decision step!

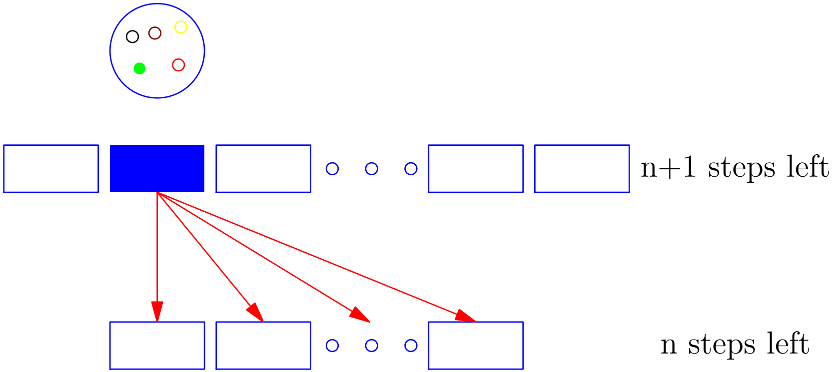


Sequential decision problems: a visual sketch

What if there are more than one next possible states?

Sequential decision problems: a visual sketch

What if there are more than one next possible states?



Sequential decision problems: a visual sketch

Sequential decision problems: a visual sketch

- The sketch gives us an intuition of SDPs but . . .

Sequential decision problems: a visual sketch

- The sketch gives us an intuition of SDPs but . . .
- . . . How can we specify a concrete SDP?

Sequential decision problems: a visual sketch

- The sketch gives us an intuition of SDPs but ...
- ... How can we specify a concrete SDP?
- ... What does it mean to solve a SDP?

Sequential decision problems: a visual sketch

- The sketch gives us an intuition of SDPs but ...
- ... How can we specify a concrete SDP?
- ... What does it mean to solve a SDP?
- ... How can we check that a solution is correct?

Mathematical specifications

Mathematical specifications

- In math $x = 1$ and $x = -1$ are said to be solutions of $x^2 = 1$.

Mathematical specifications

- In math $x = 1$ and $x = -1$ are said to be solutions of $x^2 = 1$.
- What does this precisely mean? $x = 1$, $x = -1$ and $x^2 = 1$ are all equations!

Mathematical specifications

- In math $x = 1$ and $x = -1$ are said to be solutions of $x^2 = 1$.
- What does this precisely mean? $x = 1$, $x = -1$ and $x^2 = 1$ are all equations!
- But they are in certain relations to each other. One has

  $$x = 1 \Rightarrow x^2 = 1$$

  and also

  $$x = -1 \Rightarrow x^2 = 1$$

Mathematical specifications

- In math $x = 1$ and $x = -1$ are said to be solutions of $x^2 = 1$.

- What does this precisely mean? $x = 1$, $x = -1$ and $x^2 = 1$ are all equations!

- But they are in certain relations to each other. One has

  $$x = 1 \Rightarrow x^2 = 1$$

  and also

  $$x = -1 \Rightarrow x^2 = 1$$

- These implications justify calling $x = 1$ and $x = -1$ solutions of $x^2 = 1$!

Mathematical specifications

- The equations $x = 1$ and $x = -1$ are different from $x^2 = 1$ also from another point of view.

Mathematical specifications

- The equations $x = 1$ and $x = -1$ are different from $x^2 = 1$ also from another point of view.

- They determine the value of x directly, without computations.

Mathematical specifications

- The equations $x = 1$ and $x = -1$ are different from $x^2 = 1$ also from another point of view.
- They determine the value of x directly, without computations.
- The equation $x^2 = 1$ instead specifies a problem: that of finding values whose square is one.

Mathematical specifications

- The equations $x = 1$ and $x = -1$ are different from $x^2 = 1$ also from another point of view.

- They determine the value of x directly, without computations.

- The equation $x^2 = 1$ instead specifies a problem: that of finding values whose square is one.

- We can spell out the problem a little bit more explicitly:

  Find $x \in \mathbb{R}$ **s.t.** $x^2 = 1$

Mathematical specifications

- The equations $x = 1$ and $x = -1$ are different from $x^2 = 1$ also from another point of view.
- They determine the value of x directly, without computations.
- The equation $x^2 = 1$ instead specifies a problem: that of finding values whose square is one.
- We can spell out the problem a little bit more explicitly:

  Find $x \in \mathbb{R}$ **s.t.** $x^2 = 1$

- This is a simple example of a problem specification.

Mathematical specifications

- A slightly more interesting example:

  Find $f : \mathbb{R} \to \mathbb{R}$ **s.t.** $\forall x \in \mathbb{R},\ f(x) * f(x) = x$

Mathematical specifications

- A slightly more interesting example:

  Find $f : \mathbb{R} \to \mathbb{R}$ **s.t.** $\forall x \in \mathbb{R},\ f(x) * f(x) = x$

- What does an f that fulfills the specification compute?

Mathematical specifications

- A slightly more interesting example:

  Find $f : \mathbb{R} \to \mathbb{R}$ **s.t.** $\forall x \in \mathbb{R},\ f(x) * f(x) = x$

- What does an f that fulfills the specification compute?

- Can one fulfill the specification?

Mathematical specifications

- A slightly more interesting example:

  $$\textbf{Find } f : \mathbb{R} \to \mathbb{R} \textbf{ s.t. } \forall x \in \mathbb{R},\ f(x) * f(x) = x$$

- What does an f that fulfills the specification compute?

- Can one fulfill the specification?

- We can use specifications to understand problems and give meanings to computations!

Mathematical specifications

- A slightly more interesting example:

  Find $f : \mathbb{R} \to \mathbb{R}$ **s.t.** $\forall x \in \mathbb{R},\ f(x) * f(x) = x$

- What does an f that fulfills the specification compute?

- Can one fulfill the specification?

- We can use specifications to understand problems and give meanings to computations!

- Specifications can also be applied to clarify crucial notions. We have seen that in emission problems . . .

Mathematical specifications

- ...emission reductions imply different costs and benefits for different countries.

Mathematical specifications

- . . . emission reductions imply different costs and benefits for different countries.
- The highest global benefits are obtained if most countries reduce emissions by certain (optimal, fair, ...) country-specific amounts.

Mathematical specifications

- . . . emission reductions imply different costs and benefits for different countries.
- The highest global benefits are obtained if most countries reduce emissions by certain (optimal, fair, ...) country-specific amounts.
- In this situation most countries have a free-ride opportunity.

Mathematical specifications

- ... emission reductions imply different costs and benefits for different countries.
- The highest global benefits are obtained if most countries reduce emissions by certain (optimal, fair, ...) country-specific amounts.
- In this situation most countries have a free-ride opportunity.
- The most paradigmatic example of this situation is perhaps the two-players prisoner's dilemma:

	High	*Low*
High	(1, 1)	(3,0)
Low	(0,3)	(2,2)

Mathematical specifications

- ... emission reductions imply different costs and benefits for different countries.
- The highest global benefits are obtained if most countries reduce emissions by certain (optimal, fair, ...) country-specific amounts.
- In this situation most countries have a free-ride opportunity.
- The most paradigmatic example of this situation is perhaps the two-players prisoner's dilemma:

	High	*Low*
High	$(1,1)$	(3,0)
Low	(0,3)	(2,2)

Mathematical specifications

- ...emission reductions imply different costs and benefits for different countries.
- The highest global benefits are obtained if most countries reduce emissions by certain (optimal, fair, ...) country-specific amounts.
- In this situation most countries have a free-ride opportunity.
- The most paradigmatic example of this situation is perhaps the two-players prisoner's dilemma:

	High	*Low*
High	$(1,1)$	(3,0)
Low	(0,3)	(2,2)

Mathematical specifications

- Which property makes $(High, High)$ an undesirable and yet likely outcome?

Mathematical specifications

- Which property makes $(High, High)$ an undesirable and yet likely outcome?
- What does it mean for a pair of strategies to be a Nash equilibrium?

Mathematical specifications

- Which property makes (*High*, *High*) an undesirable and yet likely outcome?
- What does it mean for a pair of strategies to be a Nash equilibrium?
- We can express this idea with a specification:

> *Let $S = \{High, Low\}$ and $p1, p2 : S \times S \to \mathbb{R}$ payoffs. A* **strategy profile** $(x, y) \in S \times S$ *is a* **Nash equilibrium** *iff* $\forall x', y' \in S$, $p1(x', y) \leqslant p1(x, y)$ *and* $p2(x, y') \leqslant p2(x, y)$.

Mathematical specifications

- Let X denote the states that a decision maker can observe.

Mathematical specifications

- Let X denote the states that a decision maker can observe.
- Let Y denote the options available to the decision maker.

Mathematical specifications

- Let X denote the states that a decision maker can observe.
- Let Y denote the options available to the decision maker.
- For simplicity, assume that Y is the same for all $x \in X$.

Mathematical specifications

- Let X denote the states that a decision maker can observe.
- Let Y denote the options available to the decision maker.
- For simplicity, assume that Y is the same for all $x \in X$.
- Policies are functions that associate an option to every state.

Mathematical specifications

- Let X denote the states that a decision maker can observe.
- Let Y denote the options available to the decision maker.
- For simplicity, assume that Y is the same for all $x \in X$.
- Policies are functions that associate an option to every state.
- Let $val : X \times Y \rightarrow Real$ be a value function: $val(x, y)$ denotes the value of taking decision y in state x.

Mathematical specifications

- Let X denote the states that a decision maker can observe.
- Let Y denote the options available to the decision maker.
- For simplicity, assume that Y is the same for all $x \in X$.
- Policies are functions that associate an option to every state.
- Let $val : X \times Y \to Real$ be a value function: $val(x, y)$ denotes the value of taking decision y in state x.
- A policy $p : X \to Y$ is called optimal w.r.t to val if it entails decisions that are better or as good as any other decision for all states.

Mathematical specifications

- Let X denote the states that a decision maker can observe.
- Let Y denote the options available to the decision maker.
- For simplicity, assume that Y is the same for all $x \in X$.
- Policies are functions that associate an option to every state.
- Let $val : X \times Y \to Real$ be a value function: $val(x, y)$ denotes the value of taking decision y in state x.
- A policy $p : X \to Y$ is called optimal w.r.t to val if it entails decisions that are better or as good as any other decision for all states.
- Exercise: give a mathematical specification of the notion of optimality for policies.

Dependent types and machine checkable specifications

- Mathematical specifications are useful to formulate notions and intentions precisely but . . .

Dependent types and machine checkable specifications

- Mathematical specifications are useful to formulate notions and intentions precisely but ...
- ... they can hardly be processed by computers.

Dependent types and machine checkable specifications

- Mathematical specifications are useful to formulate notions and intentions precisely but ...
- ... they can hardly be processed by computers.
- We want to take advantage of computers to check and perhaps even fulfill specifications!

Dependent types and machine checkable specifications

- Mathematical specifications are useful to formulate notions and intentions precisely but . . .
- . . . they can hardly be processed by computers.
- We want to take advantage of computers to check and perhaps even fulfill specifications!
- Strongly typed functional programming languages and dependent types make this possible.

Dependent types and machine checkable specifications

- Mathematical specifications are useful to formulate notions and intentions precisely but ...
- ... they can hardly be processed by computers.
- We want to take advantage of computers to check and perhaps even fulfill specifications!
- Strongly typed functional programming languages and dependent types make this possible.
- In strongly typed languages, each valid expression has a type:

$$
\begin{aligned}
1 + 2 &: \mathbb{N} \\
\texttt{"Hello"} &: \mathit{String} \\
[1, 7, 3, 8] &: \mathit{List}\ \mathbb{N}
\end{aligned}
$$

Dependent types and machine checkable specifications

- Mathematical specifications are useful to formulate notions and intentions precisely but ...
- ... they can hardly be processed by computers.
- We want to take advantage of computers to check and perhaps even fulfill specifications!
- Strongly typed functional programming languages and dependent types make this possible.
- In strongly typed languages, each valid expression has a type:

$$\begin{aligned} 1 + 2 \quad &: \mathbb{N} \\ \texttt{"Hello"} \quad &: \mathit{String} \\ [1, 7, 3, 8] \quad &: \mathit{List}\ \mathbb{N} \end{aligned}$$

- The judgment $e\ :\ t$ states that the expression e has type t.

Dependent types and machine checkable specifications

- In functional programming, one specifies what a computer shall do in terms of functions and their application and composition:

Dependent types and machine checkable specifications

- In functional programming, one specifies what a computer shall do in terms of functions and their application and composition:

  $$f \; x$$

  denotes the application of the function f to the argument x.

Dependent types and machine checkable specifications

- In functional programming, one specifies what a computer shall do in terms of functions and their application and composition:

  $$f\ x$$

  denotes the application of the function f to the argument x.

  $$g \circ f$$

  denotes the composition of g and f.

Dependent types and machine checkable specifications

- In functional programming, one specifies what a computer shall do in terms of functions and their application and composition:

  $$f\ x$$

  denotes the application of the function f to the argument x.

  $$g \circ f$$

  denotes the composition of g and f. Functions can take functions as arguments:

  $$(\circ) : (b \to c) \to (a \to b) \to a \to c$$
  $$(g \circ f)\ x = g\ (f\ x)$$

Dependent types and machine checkable specifications

- In dependently typed languages, types can depend on values:

$$
\begin{array}{l}
\textbf{data}\ \mathit{Vect}\ :\ \mathbb{N}\ \to\ \mathit{Type}\ \to\ \mathit{Type}\ \textbf{where} \\
\quad \mathit{Nil}\ :\ \mathit{Vect}\ Z\ a \\
\quad (::)\ :\ (x\ :\ a)\ \to\ (xs\ :\ \mathit{Vect}\ n\ a)\ \to\ \mathit{Vect}\ (S\ n)\ a
\end{array}
$$

Dependent types and machine checkable specifications

- In dependently typed languages, types can depend on values:

$$\begin{array}{l} \textbf{data}\ \mathit{Vect}\ :\ \mathbb{N}\ \rightarrow\ \mathit{Type}\ \rightarrow\ \mathit{Type}\ \textbf{where} \\ \quad \mathit{Nil}\ :\ \mathit{Vect}\ Z\ a \\ \quad (::)\ :\ (x\ :\ a)\ \rightarrow\ (xs\ :\ \mathit{Vect}\ n\ a)\ \rightarrow\ \mathit{Vect}\ (S\ n)\ a \end{array}$$

- Thus, we can use types to encode specifications. For instance:

$$\begin{array}{l} \mathit{Injective}\ :\ (a\ \rightarrow\ b)\ \rightarrow\ \mathit{Type} \\ \mathit{Injective}\ \{a\}\ f = (x, y\ :\ a)\ \rightarrow\ f\ x = f\ y\ \rightarrow\ x = y \end{array}$$

Dependent types and machine checkable specifications

- In dependently typed languages, types can depend on values:

$$
\begin{array}{l}
\textbf{data}\ \mathit{Vect}\ :\ \mathbb{N}\ \rightarrow\ \mathit{Type}\ \rightarrow\ \mathit{Type}\ \textbf{where} \\
\quad \mathit{Nil}\ :\ \mathit{Vect}\ Z\ a \\
\quad (::)\ :\ (x\ :\ a)\ \rightarrow\ (\mathit{xs}\ :\ \mathit{Vect}\ n\ a)\ \rightarrow\ \mathit{Vect}\ (S\ n)\ a
\end{array}
$$

- Thus, we can use types to encode specifications. For instance:

$$
\begin{array}{l}
\mathit{Injective}\ :\ (a\ \rightarrow\ b)\ \rightarrow\ \mathit{Type} \\
\mathit{Injective}\ \{a\}\ f = (x, y\ :\ a)\ \rightarrow\ f\ x = f\ y\ \rightarrow\ x = y
\end{array}
$$

- This is almost a word-by-word translation of

$$f : A \rightarrow B \textit{ injective iff } \forall x, y \in A,\ f(x) = f(y) \Rightarrow x = y.$$

Dep. types and machine checkable specifications: wrap-up

- We can apply dependent types to implement programs that are correct by construction.

Dep. types and machine checkable specifications: wrap-up

- We can apply dependent types to implement programs that are correct by construction.
- For a program P, this is done in three steps:

Dep. types and machine checkable specifications: wrap-up

- We can apply dependent types to implement programs that are correct by construction.
- For a program P, this is done in three steps:
- 1: Write a formal specification *specP* for P.

Dep. types and machine checkable specifications: wrap-up

- We can apply dependent types to implement programs that are correct by construction.
- For a program P, this is done in three steps:
- 1: Write a formal specification $specP$ for P.
- 2: Implement P.

Dep. types and machine checkable specifications: wrap-up

- We can apply dependent types to implement programs that are correct by construction.
- For a program P, this is done in three steps:
- 1: Write a formal specification $specP$ for P.
- 2: Implement P.
- 3: Implement $specP$.

Dep. types and machine checkable specifications: wrap-up

- We can apply dependent types to implement programs that are correct by construction.
- For a program P, this is done in three steps:
- 1: Write a formal specification *specP* for P.
- 2: Implement P.
- 3: Implement *specP*.
- Example of 1:

$$\begin{aligned} &P &&: \mathbb{R} \to \mathbb{R} \\ &specP &&: (x : \mathbb{R}) \to 0 \leqslant x \to (P\ x) * (P\ x) = x \end{aligned}$$

An environment for specifying and solving SDPs

- We can also apply dependent types to construct an environment for specifying and solving SDPs.

An environment for specifying and solving SDPs

- We can also apply dependent types to construct an environment for specifying and solving SDPs.
- At decision step t, $X\ t$ denotes the states the decision maker can observe:

$$X : \mathbb{N} \to \mathit{Type}$$

An environment for specifying and solving SDPs

- We can also apply dependent types to construct an environment for specifying and solving SDPs.
- At decision step t, $X\ t$ denotes the states the decision maker can observe:

$$X \;:\; \mathbb{N} \;\to\; \mathit{Type}$$

- At decision step t, $Y\ t\ x$ denotes the controls available to the decision maker in $x\ :\ X\ t$:

$$Y \;:\; (t \;:\; \mathbb{N}) \;\to\; X\ t \;\to\; \mathit{Type}$$

An environment for specifying and solving SDPs

- At decision step t, $next\ t\ x\ y$ denotes the next states that can be reached by selecting control y in the current state x:

$$next : (t : \mathbb{N}) \to (x : X\ t) \to Y\ t\ x \to M\ X\ (S\ t)$$

An environment for specifying and solving SDPs

- At decision step t, *next* $t\ x\ y$ denotes the next states that can be reached by selecting control y in the current state x:

$$next : (t : \mathbb{N}) \rightarrow (x : X\ t) \rightarrow Y\ t\ x \rightarrow M\ X\ (S\ t)$$

- Notation:
  - $S\ t = t + 1$.
  - $M = Identity \Rightarrow$ no uncertainty, deterministic SDP
  - $M = List \Rightarrow$ non-deterministic SDP
  - $M = Prob \Rightarrow$ stochastic SDP

An environment for specifying and solving SDPs

- To specify a concrete SDP, one defines X, Y and next.

An environment for specifying and solving SDPs

- To specify a concrete SDP, one defines X, Y and $next$.
- For example, in the emission problem discussed in the beginning, $X\ t$ represents the cumulated emissions, the current emissions, availability of technologies for reducing emission and a state of the world.

An environment for specifying and solving SDPs

- To specify a concrete SDP, one defines X, Y and *next*.
- For example, in the emission problem discussed in the beginning, $X\ t$ represents the cumulated emissions, the current emissions, availability of technologies for reducing emission and a state of the world.
- To complete the specification of a concrete SDP, we have to say which are the benefits that the decision maker wants to maximize:

$$\begin{aligned} &Val \ : \ Type \\ &reward \ : (t \ : \ \mathbb{N}) \ \rightarrow \ (x \ : \ X\ t) \ \rightarrow \ Y\ t\ x \ \rightarrow \ X\ (S\ t) \ \rightarrow \ Val \end{aligned}$$

An environment for specifying and solving SDPs

- In SDPs, controls are often associated with the consumption of resources: money, fuel, etc.

An environment for specifying and solving SDPs

- In SDPs, controls are often associated with the consumption of resources: money, fuel, etc.
- In our emission problem, $Y\ t\ x$ can only take two values: $High$ and Low.

An environment for specifying and solving SDPs

- In SDPs, controls are often associated with the consumption of resources: money, fuel, etc.
- In our emission problem, $Y\ t\ x$ can only take two values: *High* and *Low*.
- The decision maker seeks controls that maximize a sum of the rewards, thus values of type *Val* have to be "addable":

$$
\begin{aligned}
(\oplus) &: \mathit{Val} \rightarrow \mathit{Val} \rightarrow \mathit{Val} \\
\mathit{zero} &: \mathit{Val}
\end{aligned}
$$

An environment for specifying and solving SDPs

- In SDPs, controls are often associated with the consumption of resources: money, fuel, etc.
- In our emission problem, $Y\ t\ x$ can only take two values: *High* and *Low*.
- The decision maker seeks controls that maximize a sum of the rewards, thus values of type *Val* have to be "addable":

$$
\begin{aligned}
(\oplus) &: \mathit{Val} \rightarrow \mathit{Val} \rightarrow \mathit{Val} \\
\mathit{zero} &: \mathit{Val}
\end{aligned}
$$

- Also, we want to be able to compare benefits:

$$(\sqsubseteq) : \mathit{Val} \rightarrow \mathit{Val} \rightarrow \mathit{Type}$$

An environment for specifying and solving SDPs

- In SDPs, controls are often associated with the consumption of resources: money, fuel, etc.
- In our emission problem, $Y\ t\ x$ can only take two values: *High* and *Low*.
- The decision maker seeks controls that maximize a sum of the rewards, thus values of type *Val* have to be "addable":

$$
\begin{aligned}
(\oplus) &: \mathit{Val} \rightarrow \mathit{Val} \rightarrow \mathit{Val} \\
\mathit{zero} &: \mathit{Val}
\end{aligned}
$$

- Also, we want to be able to compare benefits:

$$
(\sqsubseteq) : \mathit{Val} \rightarrow \mathit{Val} \rightarrow \mathit{Type}
$$

- In SDPs, *Val* is often $\mathbb{N}$ or $\mathbb{R}$.

An environment for specifying and solving SDPs

- Policies are functions from states to controls:

$$
\begin{aligned}
&\mathit{Policy} \; : (t \; : \; \mathbb{N}) \; \to \; \mathit{Type} \\
&\mathit{Policy} \; t = (x \; : \; X \; t) \; \to \; Y \; t \; x
\end{aligned}
$$

An environment for specifying and solving SDPs

- Policies are functions from states to controls:

$$\begin{aligned} &\mathit{Policy} \;:\; (t \;:\; \mathbb{N}) \;\to\; \mathit{Type} \\ &\mathit{Policy}\; t = (x \;:\; X\; t) \;\to\; Y\; t\; x \end{aligned}$$

- Policy sequences are literally sequences of policies:

$$\begin{aligned} &\textbf{data}\; \mathit{PolicySeq} \;:\; (t \;:\; \mathbb{N}) \;\to\; (n \;:\; \mathbb{N}) \;\to\; \mathit{Type}\; \textbf{where} \\ &\quad \mathit{Nil} \;:\; \mathit{PolicySeq}\; t\; Z \\ &\quad (::) \;:\; \mathit{Policy}\; t \;\to\; \mathit{PolicySeq}\; (S\; t)\; n \;\to\; \mathit{PolicySeq}\; t\; (S\; n) \end{aligned}$$

An environment for specifying and solving SDPs

- Policies are functions from states to controls:

$$\begin{aligned}&\mathit{Policy} \;:\; (t \;:\; \mathbb{N}) \;\rightarrow\; \mathit{Type}\\ &\mathit{Policy}\; t = (x \;:\; X\; t) \;\rightarrow\; Y\; t\; x\end{aligned}$$

- Policy sequences are literally sequences of policies:

$$\begin{aligned}&\textbf{data}\; \mathit{PolicySeq} \;:\; (t \;:\; \mathbb{N}) \;\rightarrow\; (n \;:\; \mathbb{N}) \;\rightarrow\; \mathit{Type}\; \textbf{where}\\ &\quad \mathit{Nil} \;:\; \mathit{PolicySeq}\; t\; Z\\ &\quad (::) \;:\; \mathit{Policy}\; t \;\rightarrow\; \mathit{PolicySeq}\; (S\; t)\; n \;\rightarrow\; \mathit{PolicySeq}\; t\; (S\; n)\end{aligned}$$

- We can construct an empty policy sequence at every decision step.

An environment for specifying and solving SDPs

- Policies are functions from states to controls:

$$
\begin{aligned}
&\mathit{Policy} \ : (t \ : \ \mathbb{N}) \ \rightarrow \ \mathit{Type} \\
&\mathit{Policy} \ t = (x \ : \ X \ t) \ \rightarrow \ Y \ t \ x
\end{aligned}
$$

- Policy sequences are literally sequences of policies:

$$
\begin{aligned}
&\textbf{data} \ \mathit{PolicySeq} \ : (t \ : \ \mathbb{N}) \ \rightarrow \ (n \ : \ \mathbb{N}) \ \rightarrow \ \mathit{Type} \ \textbf{where} \\
&\quad \mathit{Nil} \ : \ \mathit{PolicySeq} \ t \ Z \\
&\quad (::) \ : \ \mathit{Policy} \ t \ \rightarrow \ \mathit{PolicySeq} \ (S \ t) \ n \ \rightarrow \ \mathit{PolicySeq} \ t \ (S \ n)
\end{aligned}
$$

- We can construct an empty policy sequence at every decision step.
- With a decision policy for step t and a sequence of n policies, we can construct a policy sequence for $n+1$ decision steps.

An environment for specifying and solving SDPs

- We can compute the value of taking n decisions according to a policy sequence in terms of a sum of the rewards obtained:

$$
\begin{array}{l}
val : (x : X\ t) \to PolicySeq\ t\ n \to Val \\
val\ x\ Nil = zero \\
val\ x\ (p :: ps) = reward\ t\ x\ y\ x' \oplus val\ x'\ ps\ \textbf{where} \\
\quad y\ \ : Y\ t\ x \\
\quad y\ \ = p\ x \\
\quad x'\ : X\ (S\ t) \\
\quad x' = next\ t\ x\ y
\end{array}
$$

An environment for specifying and solving SDPs

- We can compute the value of taking n decisions according to a policy sequence in terms of a sum of the rewards obtained:

$$
\begin{aligned}
&val \;:\; (x \;:\; X\ t) \;\to\; PolicySeq\ t\ n \;\to\; Val \\
&val\ x\ Nil = zero \\
&val\ x\ (p :: ps) = reward\ t\ x\ y\ x' \oplus val\ x'\ ps\ \textbf{where} \\
&\quad y \;:\; Y\ t\ x \\
&\quad y \;= p\ x \\
&\quad x' \;:\; X\ (S\ t) \\
&\quad x' = next\ t\ x\ y
\end{aligned}
$$

- Remember that the decision maker seeks controls that maximize a sum of the rewards.

An environment for specifying and solving SDPs

- We can compute the value of taking n decisions according to a policy sequence in terms of a sum of the rewards obtained:

$$
\begin{aligned}
& \mathit{val} \;:\; (x \;:\; X\;t) \;\to\; \mathit{PolicySeq}\;t\;n \;\to\; \mathit{Val} \\
& \mathit{val}\;x\;\mathit{Nil} = \mathit{zero} \\
& \mathit{val}\;x\;(p :: ps) = \mathit{reward}\;t\;x\;y\;x' \oplus \mathit{val}\;x'\;ps\;\textbf{where} \\
& \quad y \;\;:\; Y\;t\;x \\
& \quad y \;\;= p\;x \\
& \quad x' \;:\; X\;(S\;t) \\
& \quad x' = \mathit{next}\;t\;x\;y
\end{aligned}
$$

- Remember that the decision maker seeks controls that maximize a sum of the rewards.
- *val* computes precisely such a sum for arbitrary policy sequences!

An environment for specifying and solving SDPs

- Thus, we can use *val* to express what it means for a policy sequence to be optimal:

$$
\begin{aligned}
&\mathit{OptPolicySeq} \;:\; \mathit{PolicySeq}\; t\; n \;\rightarrow\; \mathit{Type} \\
&\mathit{OptPolicySeq}\; ps = (x \;:\; X\; t) \;\rightarrow\; (ps' \;:\; \mathit{PolicySeq}\; t\; n) \;\rightarrow \\
&\qquad\qquad\qquad\qquad \mathit{val}\; x\; ps' \sqsubseteq \mathit{val}\; x\; ps
\end{aligned}
$$

An environment for specifying and solving SDPs

- Thus, we can use *val* to express what it means for a policy sequence to be optimal:

$$\begin{aligned}
&\mathit{OptPolicySeq} \;:\; \mathit{PolicySeq}\ t\ n \;\to\; \mathit{Type}\\
&\mathit{OptPolicySeq}\ \mathit{ps} = (x \;:\; X\ t) \;\to\; (\mathit{ps}' \;:\; \mathit{PolicySeq}\ t\ n) \;\to\\
&\qquad\qquad\qquad\qquad \mathit{val}\ x\ \mathit{ps}' \sqsubseteq \mathit{val}\ x\ \mathit{ps}
\end{aligned}$$

- But ...

An environment for specifying and solving SDPs

- Thus, we can use *val* to express what it means for a policy sequence to be optimal:

$$
\begin{aligned}
&\mathit{OptPolicySeq} \;:\; \mathit{PolicySeq}\ t\ n \;\rightarrow\; \mathit{Type} \\
&\mathit{OptPolicySeq}\ \mathit{ps} = (x \;:\; X\ t) \;\rightarrow\; (\mathit{ps}' \;:\; \mathit{PolicySeq}\ t\ n) \;\rightarrow \\
&\qquad\qquad\qquad\qquad \mathit{val}\ x\ \mathit{ps}' \sqsubseteq \mathit{val}\ x\ \mathit{ps}
\end{aligned}
$$

- But ...

- ... How do we compute optimal policies?

An environment for specifying and solving SDPs

- The key idea lies in the notion of optimal extensions.

An environment for specifying and solving SDPs

- The key idea lies in the notion of optimal extensions.

- A policy p is an optimal extension of a policy sequence ps if there is no better way to make one more decision steps than $p :: ps$:

An environment for specifying and solving SDPs

- The key idea lies in the notion of optimal extensions.

- A policy p is an optimal extension of a policy sequence ps if there is no better way to make one more decision steps than $p :: ps$:

$$
\begin{aligned}
&OptExt \;:\; PolicySeq\;(S\;t)\;m \;\rightarrow\; Policy\;t \;\rightarrow\; Type \\
&OptExt\;ps\;p = (x \;:\; X\;t) \;\rightarrow\; (p' \;:\; Policy\;t) \;\rightarrow \\
&\qquad\qquad\qquad val\;x\;(p' :: ps) \sqsubseteq val\;x\;(p :: ps)
\end{aligned}
$$

An environment for specifying and solving SDPs

- Optimal extensions are the key for computing optimal policy sequences because of Bellman's optimality principle:

An environment for specifying and solving SDPs

- Optimal extensions are the key for computing optimal policy sequences because of Bellman's optimality principle:

$$\begin{array}{lll} \mathit{Bellman} : & (\mathit{ps} \; : \; \mathit{PolicySeq} \; (S \; t) \; m) & \to \; \mathit{OptPolicySeq} \; \mathit{ps} \; \to \\ & (p \;\; : \; \mathit{Policy} \; t) & \to \; \mathit{OptExt} \; \mathit{ps} \; p \; \to \\ & \mathit{OptPolicySeq} \; (p :: \mathit{ps}) & \end{array}$$

An environment for specifying and solving SDPs

- Optimal extensions are the key for computing optimal policy sequences because of Bellman's optimality principle:

$$
\begin{aligned}
\mathit{Bellman} \; : \; & (\mathit{ps} \; : \; \mathit{PolicySeq}\;(S\;t)\;m) \;\rightarrow\; \mathit{OptPolicySeq}\;\mathit{ps} \;\rightarrow \\
& (p \quad : \; \mathit{Policy}\;t) \qquad\qquad\quad \rightarrow\; \mathit{OptExt}\;\mathit{ps}\;p \;\rightarrow \\
& \mathit{OptPolicySeq}\;(p :: \mathit{ps})
\end{aligned}
$$

- Proving *Bellman* is not difficult but a little bit technical.

An environment for specifying and solving SDPs

- Optimal extensions are the key for computing optimal policy sequences because of Bellman's optimality principle:

$$\begin{aligned} \mathit{Bellman} \ : \ & (\mathit{ps} \ : \ \mathit{PolicySeq}\ (S\ t)\ m) \ \to \ \mathit{OptPolicySeq}\ \mathit{ps} \ \to \\ & (p \ \ : \ \mathit{Policy}\ t) \qquad\qquad\quad \to \ \mathit{OptExt}\ \mathit{ps}\ p \ \to \\ & \mathit{OptPolicySeq}\ (p :: \mathit{ps}) \end{aligned}$$

- Proving *Bellman* is not difficult but a little bit technical.

- Instead of proving the result, we are going to apply it!

An environment for specifying and solving SDPs

- Assume that we have a method for computing optimal extensions of arbitrary policy sequences:

An environment for specifying and solving SDPs

- Assume that we have a method for computing optimal extensions of arbitrary policy sequences:

$$optExt \ : \ PolicySeq \ (S \ t) \ n \ \rightarrow \ Policy \ t$$

$$\begin{aligned} optExtSpec \ : \ & (ps \ : \ PolicySeq \ (S \ t) \ n) \ \rightarrow \\ & OptExt \ ps \ (optExt \ ps) \end{aligned}$$

An environment for specifying and solving SDPs

- Assume that we have a method for computing optimal extensions of arbitrary policy sequences:

$$optExt \;:\; PolicySeq\;(S\;t)\;n \;\to\; Policy\;t$$

$$\begin{aligned} optExtSpec \;:\; & (ps \;:\; PolicySeq\;(S\;t)\;n) \;\to \\ & OptExt\;ps\;(optExt\;ps) \end{aligned}$$

- Then

$$\begin{aligned}
& backwardsInduction \;:\; (t \;:\; \mathbb{N}) \;\to\; (n \;:\; \mathbb{N}) \;\to\; PolicySeq\;t\;n \\
& backwardsInduction\;t\;Z \qquad = Nil \\
& backwardsInduction\;t\;(S\;n) = optExt\;ps :: ps\;\mathbf{where} \\
& \quad ps \;:\; PolicySeq\;(S\;t)\;n \\
& \quad ps = backwardsInduction\;(S\;t)\;n
\end{aligned}$$

computes optimal policy sequences for arbitrary SDPs!

An environment for specifying and solving SDPs

- Assume that we have a method for computing optimal extensions of arbitrary policy sequences:

$$optExt \ : \ PolicySeq \ (S \ t) \ n \ \to \ Policy \ t$$

$$\begin{aligned} optExtSpec \ : \ & (ps \ : \ PolicySeq \ (S \ t) \ n) \ \to \\ & OptExt \ ps \ (optExt \ ps) \end{aligned}$$

- Then

$$\begin{aligned}
&backwardsInduction \ : (t \ : \ \mathbb{N}) \ \to \ (n \ : \ \mathbb{N}) \ \to \ PolicySeq \ t \ n \\
&backwardsInduction \ t \ Z \qquad = Nil \\
&backwardsInduction \ t \ (S \ n) = optExt \ ps :: ps \ \textbf{where} \\
&\quad ps \ : \ PolicySeq \ (S \ t) \ n \\
&\quad ps = backwardsInduction \ (S \ t) \ n
\end{aligned}$$

computes optimal policy sequences for arbitrary SDPs! Proof:

An environment for specifying and solving SDPs

$$\begin{aligned}
\mathit{backwardsInductionLemma} \; : \; (t \; : \; \mathbb{N}) \; \rightarrow \; (n \; : \; \mathbb{N}) \; \rightarrow \\
\mathit{OptPolicySeq} \; (\mathit{backwardsInduction} \; t \; n)
\end{aligned}$$

An environment for specifying and solving SDPs

$$
\begin{aligned}
&\mathit{backwardsInductionLemma} : (t : \mathbb{N}) \rightarrow (n : \mathbb{N}) \rightarrow \\
&\qquad\qquad\qquad\qquad\qquad \mathit{OptPolicySeq}\ (\mathit{backwardsInduction}\ t\ n)
\end{aligned}
$$

$$
\begin{aligned}
&\mathit{backwardsInductionLemma}\ t\ (S\ n) = \mathit{Bellman}\ \mathit{ps}\ \mathit{ops}\ p\ \mathit{oep}\ \textbf{where} \\
&\quad \mathit{ps} \ \ : \mathit{PolicySeq}\ (S\ t)\ n \\
&\quad \mathit{ps} \ \ = \mathit{backwardsInduction}\ (S\ t)\ n \\
&\quad \mathit{ops} : \mathit{OptPolicySeq}\ \mathit{ps} \\
&\quad \mathit{ops} = \mathit{backwardsInductionLemma}\ (S\ t)\ n \\
&\quad p \ \ \ : \mathit{Policy}\ t \\
&\quad p \ \ \ = \mathit{optExt}\ \mathit{ps} \\
&\quad \mathit{oep} : \mathit{OptExt}\ \mathit{ps}\ p \\
&\quad \mathit{oep} = \mathit{optExtSpec}\ \mathit{ps}
\end{aligned}
$$

An environment for specifying and solving SDPs: wrap-up

- We have built an environment to specify and solve arbitrary SDPs.

An environment for specifying and solving SDPs: wrap-up

- We have built an environment to specify and solve arbitrary SDPs.
- The method yields provably optimal policies.

An environment for specifying and solving SDPs: wrap-up

- We have built an environment to specify and solve arbitrary SDPs.
- The method yields provably optimal policies.
- Thus, we can provide decision makers with rigorous rules to take best decisions under uncertainty and imperfect information!

An environment for specifying and solving SDPs: wrap-up

- We have built an environment to specify and solve arbitrary SDPs.
- The method yields provably optimal policies.
- Thus, we can provide decision makers with rigorous rules to take best decisions under uncertainty and imperfect information!
- Two questions remain open:

An environment for specifying and solving SDPs: wrap-up

- We have built an environment to specify and solve arbitrary SDPs.
- The method yields provably optimal policies.
- Thus, we can provide decision makers with rigorous rules to take best decisions under uncertainty and imperfect information!
- Two questions remain open:
- 1: When and how can we compute optimal extensions?

An environment for specifying and solving SDPs: wrap-up

- We have built an environment to specify and solve arbitrary SDPs.
- The method yields provably optimal policies.
- Thus, we can provide decision makers with rigorous rules to take best decisions under uncertainty and imperfect information!
- Two questions remain open:
- 1: When and how can we compute optimal extensions?
- 2: How do we apply the method?

How do we apply the method?

Controls

Controls

- At each decision step the decision maker has only two options: low or high emissions:

Controls

- At each decision step the decision maker has only two options: low or high emissions:

$$Y\ t\ x = \{Low, High\}$$

Controls

- At each decision step the decision maker has only two options: low or high emissions:

$$Y\ t\ x = \{Low, High\}$$

- Low emissions, if implemented, increase the cumulated emissions less than high emissions.

Controls

- At each decision step the decision maker has only two options: low or high emissions:

$$Y\ t\ x = \{Low, High\}$$

- Low emissions, if implemented, increase the cumulated emissions less than high emissions.

- Without lost of generality, we can take these increases to be zero and one.

States

States

- At each step, the decision maker has to choose between low and high emissions on the basis of four data:

States

- At each step, the decision maker has to choose between low and high emissions on the basis of four data:
  - The amount of cumulated emissions.

States

- At each step, the decision maker has to choose between low and high emissions on the basis of four data:
  - The amount of cumulated emissions.
  - The current emission level $E = \{Low, High\}$.

States

- At each step, the decision maker has to choose between low and high emissions on the basis of four data:
  - The amount of cumulated emissions.
  - The current emission level $E = \{Low, High\}$.
  - The availability of effective technologies for reducing emissions $T = \{Available, Unavailable\}$.

States

- At each step, the decision maker has to choose between low and high emissions on the basis of four data:
  - The amount of cumulated emissions.
  - The current emission level $E = \{Low, High\}$.
  - The availability of effective technologies for reducing emissions $T = \{Available, Unavailable\}$.
  - A state of the world $W = \{Good, Bad\}$.

States

- At each step, the decision maker has to choose between low and high emissions on the basis of four data:
  - The amount of cumulated emissions.
  - The current emission level $E = \{\mathit{Low}, \mathit{High}\}$.
  - The availability of effective technologies for reducing emissions $T = \{\mathit{Available}, \mathit{Unavailable}\}$.
  - A state of the world $W = \{\mathit{Good}, \mathit{Bad}\}$.
- Thus, states are just tuples of 4 values:

$$X\ t = (\{0\,..\,t\}, E, T, W)$$

Uncertainties

Uncertainties

- The process starts with zero cumulated emissions, high emissions, unavailable technologies and in a good world

Uncertainties

- The process starts with zero cumulated emissions, high emissions, unavailable technologies and in a good world
- In this state, the probability that the world turns bad is low.

Uncertainties

- The process starts with zero cumulated emissions, high emissions, unavailable technologies and in a good world
- In this state, the probability that the world turns bad is low.
- But if the cumulated emissions increase beyond a critical threshold $crE : \mathbb{R}$, the probability that the world becomes bad increases.

Uncertainties

- The process starts with zero cumulated emissions, high emissions, unavailable technologies and in a good world
- In this state, the probability that the world turns bad is low.
- But if the cumulated emissions increase beyond a critical threshold $crE : \mathbb{R}$, the probability that the world becomes bad increases.
- Once the world has reached a bad state, there is no chance to turn back to a good state.

Uncertainties

- The process starts with zero cumulated emissions, high emissions, unavailable technologies and in a good world
- In this state, the probability that the world turns bad is low.
- But if the cumulated emissions increase beyond a critical threshold $crE : \mathbb{R}$, the probability that the world becomes bad increases.
- Once the world has reached a bad state, there is no chance to turn back to a good state.
- Similarly, the probability that effective technologies become available is low in the beginning and increases after a critical number of decision steps $crN : \mathbb{N}$.

Uncertainties

- The process starts with zero cumulated emissions, high emissions, unavailable technologies and in a good world
- In this state, the probability that the world turns bad is low.
- But if the cumulated emissions increase beyond a critical threshold $crE : \mathbb{R}$, the probability that the world becomes bad increases.
- Once the world has reached a bad state, there is no chance to turn back to a good state.
- Similarly, the probability that effective technologies become available is low in the beginning and increases after a critical number of decision steps $crN : \mathbb{N}$.
- Once available, effective technologies stay available for ever.

Uncertainties

Uncertainties

- Low/high emission decisions may or may not be implemented.

Uncertainties

- Low/high emission decisions may or may not be implemented.

- After *crN* decision steps, efficient technologies may or may not become available.

Uncertainties

- Low/high emission decisions may or may not be implemented.
- After *crN* decision steps, efficient technologies may or may not become available.
- Exceeding *crE* may or may not turn the world into a bad state.

Uncertainties

Uncertainties

- Low/high emission decisions may or may not be implemented:

Uncertainties

- Low/high emission decisions may or may not be implemented:
  - pLL – the probability of implementing low emissions when the current emissions measures are low and low emissions are chosen.

Uncertainties

- Low/high emission decisions may or may not be implemented:
  - pLL – the probability of implementing low emissions when the current emissions measures are low and low emissions are chosen.
  - pLH – the probability of implementing low emissions when the current emissions measures are high and low emissions are chosen.

Uncertainties

- Low/high emission decisions may or may not be implemented:
  - pLL – the probability of implementing low emissions when the current emissions measures are low and low emissions are chosen.
  - pLH – the probability of implementing low emissions when the current emissions measures are high and low emissions are chosen.
  - pHL – the probability of implementing high emissions when the current emissions measures are low and high emissions are chosen.

Uncertainties

- Low/high emission decisions may or may not be implemented:
  - *pLL* – the probability of implementing low emissions when the current emissions measures are low and low emissions are chosen.
  - *pLH* – the probability of implementing low emissions when the current emissions measures are high and low emissions are chosen.
  - *pHL* – the probability of implementing high emissions when the current emissions measures are low and high emissions are chosen.
  - *pHH* – the probability of implementing high emissions when the current emissions measures are high and high emissions are chosen.

Uncertainties

- Low/high emission decisions may or may not be implemented:
  - pLL – the probability of implementing low emissions when the current emissions measures are low and low emissions are chosen.
  - pLH – the probability of implementing low emissions when the current emissions measures are high and low emissions are chosen.
  - pHL – the probability of implementing high emissions when the current emissions measures are low and high emissions are chosen.
  - pHH – the probability of implementing high emissions when the current emissions measures are high and high emissions are chosen.
- Constraints: $pLH \leqslant pLL$ and $pHL \leqslant pHH$.

Uncertainties

- After *crN* decision steps, efficient technologies may or may not become available:

Uncertainties

- After *crN* decision steps, efficient technologies may or may not become available:
  - *pA1* – the probability that effective technologies become available when the number of decision steps is $\leqslant$ *crN*,

Uncertainties

- After *crN* decision steps, efficient technologies may or may not become available:
  - *pA1* – the probability that effective technologies become available when the number of decision steps is $\leqslant crN$,
  - *pA2* – the probability that effective technologies become available when the number of decision steps is $> crN$.

Uncertainties

- After *crN* decision steps, efficient technologies may or may not become available:
  - *pA1* – the probability that effective technologies become available when the number of decision steps is $\leqslant crN$,
  - *pA2* – the probability that effective technologies become available when the number of decision steps is $> crN$.
- Constraint: $pA1 \leqslant pA2$.

Uncertainties

- Exceeding *crE* may or may not turn the world into a bad state:

Uncertainties

- Exceeding *crE* may or may not turn the world into a bad state:
  - *pS1* – the probability of staying in a good world when the cumulated emissions are $\leqslant crE$,

Uncertainties

- Exceeding *crE* may or may not turn the world into a bad state:
  - *pS1* – the probability of staying in a good world when the cumulated emissions are $\leqslant$ *crE*,
  - *pS2* – the probability of staying in a good world when the cumulated emissions are $>$ *crE*.

Uncertainties

- Exceeding *crE* may or may not turn the world into a bad state:
  - *pS1* – the probability of staying in a good world when the cumulated emissions are $\leqslant$ *crE*,
  - *pS2* – the probability of staying in a good world when the cumulated emissions are $>$ *crE*.
- Constraint: *pS2* $\leqslant$ *pS1*.

Transition function

Transition function

- Given crN, crE and the probabilities $pLL \ldots pS2$, the transition function can be defined by cases.

Transition function

- Given crN, crE and the probabilities $pLL \ldots pS2$, the transition function can be defined by cases.
- For instance, if
  - the current state is $x = (e, H, U, G)$,

Transition function

- Given crN, crE and the probabilities $pLL \ldots pS2$, the transition function can be defined by cases.
- For instance, if
  - the current state is $x = (e, H, U, G)$,
  - the decision maker has opted for low emissions,

Transition function

- Given *crN*, *crE* and the probabilities *pLL* ... *pS2*, the transition function can be defined by cases.
- For instance, if
  - the current state is $x = (e, H, U, G)$,
  - the decision maker has opted for low emissions,
  - $e \leqslant crE$ and

Transition function

- Given crN, crE and the probabilities $pLL \ldots pS2$, the transition function can be defined by cases.
- For instance, if
  - the current state is $x = (e, H, U, G)$,
  - the decision maker has opted for low emissions,
  - $e \leqslant crE$ and
  - $t \leqslant crN \ldots$

Transition function

- the result of $next\ t\ x\ L$ is a probability distribution with the assignments:

Transition function

- the result of *next t x L* is a probability distribution with the assignments:

$$
\begin{array}{lcl}
\textit{prob}\,(e, \quad L, U, G) & = & pLH \;*\; (1 - pA1) \;*\; pS1 \\
\textit{prob}\,(e+1, H, U, G) & = & (1 - pLH) \;*\; (1 - pA1) \;*\; pS1 \\
\textit{prob}\,(e, \quad L, A, G) & = & pLH \;*\; pA1 \;*\; pS1 \\
\textit{prob}\,(e+1, H, A, G) & = & (1 - pLH) \;*\; pA1 \;*\; pS1 \\
\textit{prob}\,(e, \quad L, U, B) & = & pLH \;*\; (1 - pA1) \;*\; (1 - pS1) \\
\textit{prob}\,(e+1, H, U, B) & = & (1 - pLH) \;*\; (1 - pA1) \;*\; (1 - pS1) \\
\textit{prob}\,(e, \quad L, A, B) & = & pLH \;*\; pA1 \;*\; (1 - pS1) \\
\textit{prob}\,(e+1, H, A, B) & = & (1 - pLH) \;*\; pA1 \;*\; (1 - pS1)
\end{array}
$$

Rewards

Rewards

The idea is that:

- Being in a bad world yields less benefits (more damages) than being in a good world.

Rewards

The idea is that:

- Being in a bad world yields less benefits (more damages) than being in a good world.
- Low emissions yield less benefits (more costs, less growth) than high emissions.

Rewards

The idea is that:

- Being in a bad world yields less benefits (more damages) than being in a good world.
- Low emissions yield less benefits (more costs, less growth) than high emissions.
- Implementing low emissions when effective technologies are unavailable costs more than implementing emissions when these technologies are available.

Rewards

Without loss of generality, we can take the benefits of being in a good world for a step to be one and define

$$
\begin{aligned}
\mathit{reward}\ t\ x\ y\ (e, H, U, G) &= 1 + h \\
\mathit{reward}\ t\ x\ y\ (e, H, U, B) &= b + h \\
\mathit{reward}\ t\ x\ y\ (e, H, A, G) &= 1 + h \\
\mathit{reward}\ t\ x\ y\ (e, H, A, B) &= b + h \\
\mathit{reward}\ t\ x\ y\ (e, L, U, G) &= 1 + \mathit{lu} \\
\mathit{reward}\ t\ x\ y\ (e, L, U, B) &= b + \mathit{lu} \\
\mathit{reward}\ t\ x\ y\ (e, L, A, G) &= 1 + \mathit{la} \\
\mathit{reward}\ t\ x\ y\ (e, L, A, B) &= b + \mathit{la}
\end{aligned}
$$

where $h, b, \mathit{lu}, \mathit{la} : \mathbb{R}$ fulfil $b \leqslant 1$, $0 \leqslant \mathit{lu} \leqslant \mathit{la} \leqslant h$.

Rewards

Without loss of generality, we can take the benefits of being in a good world for a step to be one and define

$$\begin{aligned}
reward\ t\ x\ y\ (e, H, U, G) &= 1 + h \\
reward\ t\ x\ y\ (e, H, U, B) &= b + h \\
reward\ t\ x\ y\ (e, H, A, G) &= 1 + h \\
reward\ t\ x\ y\ (e, H, A, B) &= b + h \\
reward\ t\ x\ y\ (e, L, U, G) &= 1 + lu \\
reward\ t\ x\ y\ (e, L, U, B) &= b + lu \\
reward\ t\ x\ y\ (e, L, A, G) &= 1 + la \\
reward\ t\ x\ y\ (e, L, A, B) &= b + la
\end{aligned}$$

where $h, b, lu, la : \mathbb{R}$ fulfil $b \leqslant 1$, $0 \leqslant lu \leqslant la \leqslant h$. Notice that

- The minimal cost of implementing low emissions is $h - la$

Rewards

Without loss of generality, we can take the benefits of being in a good world for a step to be one and define

$$
\begin{aligned}
\mathit{reward}\ t\ x\ y\ (e, H, U, G) &= 1 + h \\
\mathit{reward}\ t\ x\ y\ (e, H, U, B) &= b + h \\
\mathit{reward}\ t\ x\ y\ (e, H, A, G) &= 1 + h \\
\mathit{reward}\ t\ x\ y\ (e, H, A, B) &= b + h \\
\mathit{reward}\ t\ x\ y\ (e, L, U, G) &= 1 + \mathit{lu} \\
\mathit{reward}\ t\ x\ y\ (e, L, U, B) &= b + \mathit{lu} \\
\mathit{reward}\ t\ x\ y\ (e, L, A, G) &= 1 + \mathit{la} \\
\mathit{reward}\ t\ x\ y\ (e, L, A, B) &= b + \mathit{la}
\end{aligned}
$$

where $h, b, \mathit{lu}, \mathit{la} : \mathbb{R}$ fulfil $b \leqslant 1$, $0 \leqslant \mathit{lu} \leqslant \mathit{la} \leqslant h$. Notice that

- The minimal cost of implementing low emissions is $h - \mathit{la}$
- The step costs of being in a bad world are $1 - b$

Rewards

Without loss of generality, we can take the benefits of being in a good world for a step to be one and define

$$
\begin{aligned}
reward\ t\ x\ y\ (e, H, U, G) &= 1 + h \\
reward\ t\ x\ y\ (e, H, U, B) &= b + h \\
reward\ t\ x\ y\ (e, H, A, G) &= 1 + h \\
reward\ t\ x\ y\ (e, H, A, B) &= b + h \\
reward\ t\ x\ y\ (e, L, U, G) &= 1 + lu \\
reward\ t\ x\ y\ (e, L, U, B) &= b + lu \\
reward\ t\ x\ y\ (e, L, A, G) &= 1 + la \\
reward\ t\ x\ y\ (e, L, A, B) &= b + la
\end{aligned}
$$

where $h, b, lu, la\ :\ \mathbb{R}$ fulfil $b \leqslant 1$, $0 \leqslant lu \leqslant la \leqslant h$. Notice that

- The minimal cost of implementing low emissions is $h - la$
- The step costs of being in a bad world are $1 - b$
- $1 - b < h - la \Rightarrow$ reducing emissions is never a best choice!

Completing the specification

Completing the specification

- *State*, *Ctrl*, *next* and *reward* as discussed.

Completing the specification

- *State*, *Ctrl*, *next* and *reward* as discussed.
- $M = \mathit{Prob}$, $\mathit{Val} = \mathbb{R}$, $\sqsubseteq = \mathit{lift} \leqslant$, $\oplus = +$, *meas* is the expected value.

Completing the specification

- *State*, *Ctrl*, *next* and *reward* as discussed.
- $M = Prob$, $Val = \mathbb{R}$, $\sqsubseteq = lift \leqslant$, $\oplus = +$, *meas* is the expected value.
- 9 decision steps, starting in $(0, H, U, G)$: zero cumulated emissions, high emissions, unavailable efficient technologies and world in a good state.

Completing the specification

- *State*, *Ctrl*, *next* and *reward* as discussed.
- $M = \mathit{Prob}$, $\mathit{Val} = \mathbb{R}$, $\sqsubseteq = \mathit{lift} \leqslant$, $\oplus = +$, *meas* is the expected value.
- 9 decision steps, starting in $(0, H, U, G)$: zero cumulated emissions, high emissions, unavailable efficient technologies and world in a good state.
- $\mathit{crE} = 4$ and $\mathit{crN} = 2$.

Completing the specification

- *State*, *Ctrl*, *next* and *reward* as discussed.
- $M = Prob$, $Val = \mathbb{R}$, $\sqsubseteq = lift \leqslant$, $\oplus = +$, *meas* is the expected value.
- 9 decision steps, starting in $(0, H, U, G)$: zero cumulated emissions, high emissions, unavailable efficient technologies and world in a good state.
- $crE = 4$ and $crN = 2$.
- $b = 0.5$, $lu = 0.1$, $la = 0.2$, $h = 0.3$.

Logical results

Logical results

- If the state of the world is bad, reducing emissions can never be optimal: Après moi le déluge.

Logical results

- If the state of the world is bad, reducing emissions can never be optimal: Après moi le déluge.
- Conversely, reducing emissions can only pay off if it allows avoiding transitions to a bad world.

Logical results

- If the state of the world is bad, reducing emissions can never be optimal: Après moi le déluge.
- Conversely, reducing emissions can only pay off if it allows avoiding transitions to a bad world.
- At the last decision step it is always optimal to select high emissions.

Logical results

- If the state of the world is bad, reducing emissions can never be optimal: Après moi le déluge.
- Conversely, reducing emissions can only pay off if it allows avoiding transitions to a bad world.
- At the last decision step it is always optimal to select high emissions.
- *crE* $= 4 \Rightarrow$ it takes at least 5 steps to achieve states in which the sum of the cumulated emissions exceeds *crE* and, the probability of a transition to a bad world increases from *pS1* to *pS2*.

Logical results

- If the state of the world is bad, reducing emissions can never be optimal: Après moi le déluge.
- Conversely, reducing emissions can only pay off if it allows avoiding transitions to a bad world.
- At the last decision step it is always optimal to select high emissions.
- *crE* $= 4 \Rightarrow$ it takes at least 5 steps to achieve states in which the sum of the cumulated emissions exceeds *crE* and, the probability of a transition to a bad world increases from *pS1* to *pS2*.
- *crN* $= 2 \Rightarrow$ it takes 3 steps to achieve states in which the probability that effective technologies for reducing GHG emissions become available increases from *pA1* to *pA2*.

Certain case: basic facts

Certain case: basic facts

- $pS2 = pA1 = 0$, $pS1 = pA2 = pLL = pLH = pHL = pHH = 1$.

Certain case: basic facts

- $pS2 = pA1 = 0$, $pS1 = pA2 = pLL = pLH = pHL = pHH = 1$.
- Effective technologies become available (with certainty) after 4 steps.

Certain case: basic facts

- $pS2 = pA1 = 0$, $pS1 = pA2 = pLL = pLH = pHL = pHH = 1$.
- Effective technologies become available (with certainty) after 4 steps.
- The state of the world turns bad (with certainty) after 6 steps at high emissions.

Certain case: basic facts

- $pS2 = pA1 = 0$, $pS1 = pA2 = pLL = pLH = pHL = pHH = 1$.
- Effective technologies become available (with certainty) after 4 steps.
- The state of the world turns bad (with certainty) after 6 steps at high emissions.
- For any given policy sequence there is exactly one possible state-control trajectory.

Certain case: policies

- *Const High* policies:

Certain case: policies

- *Const High* policies:
  - trajectories, probabilities, rewards:

```
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )], 100%, 9.7
```

Certain case: policies

- *Const High* policies:
  - trajectories, probabilities, rewards:
    ```
    [((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
     ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )], 100%, 9.7
    ```
  - Expected sum of rewards = 9.7.

Certain case: policies

- *Const High* policies:
  - trajectories, probabilities, rewards:

    ```
    [((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
     ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )], 100%, 9.7
    ```
  - Expected sum of rewards = 9.7.
- *Const Low* policies:

Certain case: policies

- *Const High* policies:
  - trajectories, probabilities, rewards:
    ```
    [((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
     ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )], 100%, 9.7
    ```
  - Expected sum of rewards = 9.7.
- *Const Low* policies:
  - trajectories, probabilities, rewards
    ```
    [((0,H,U,G),L), ((0,L,U,G),L), ((0,L,U,G),L), ((0,L,U,G),L), ((0,L,A,G),L),
     ((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G), )], 100%, 10.5
    ```

Certain case: policies

- *Const High* policies:
  - trajectories, probabilities, rewards:

```
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )], 100%, 9.7
```

  - Expected sum of rewards = 9.7.
- *Const Low* policies:
  - trajectories, probabilities, rewards

```
[((0,H,U,G),L), ((0,L,U,G),L), ((0,L,U,G),L), ((0,L,U,G),L), ((0,L,A,G),L),
 ((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G), )], 100%, 10.5
```

  - Expected sum of rewards = 10.5

Certain case: policies

- *Const High* policies:
  - trajectories, probabilities, rewards:
    `[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),`
    `((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )], 100%, 9.7`
  - Expected sum of rewards = 9.7.
- *Const Low* policies:
  - trajectories, probabilities, rewards
    `[((0,H,U,G),L), ((0,L,U,G),L), ((0,L,U,G),L), ((0,L,U,G),L), ((0,L,A,G),L),`
    `((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G), )], 100%, 10.5`
  - Expected sum of rewards = 10.5
- Optimal policies:

Certain case: policies

- *Const High* policies:
  - trajectories, probabilities, rewards:

    ```
    [((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
     ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )], 100%, 9.7
    ```

  - Expected sum of rewards = 9.7.
- *Const Low* policies:
  - trajectories, probabilities, rewards

    ```
    [((0,H,U,G),L), ((0,L,U,G),L), ((0,L,U,G),L), ((0,L,U,G),L), ((0,L,A,G),L),
     ((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G), )], 100%, 10.5
    ```

  - Expected sum of rewards = 10.5
- Optimal policies:
  - trajectories, probabilities, rewards

    ```
    [((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),L),
     ((4,L,A,G),L), ((4,L,A,G),L), ((4,L,A,G),L), ((4,L,A,G),H), ((5,H,A,G)  )], 100%, 11.3
    ```

Certain case: policies

- *Const High* policies:
  - trajectories, probabilities, rewards:
    `[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),`
    `((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )], 100%, 9.7`
  - Expected sum of rewards = 9.7.
- *Const Low* policies:
  - trajectories, probabilities, rewards
    `[((0,H,U,G),L), ((0,L,U,G),L), ((0,L,U,G),L), ((0,L,U,G),L), ((0,L,A,G),L),`
    `((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G), )], 100%, 10.5`
  - Expected sum of rewards = 10.5
- Optimal policies:
  - trajectories, probabilities, rewards
    `[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),L),`
    `((4,L,A,G),L), ((4,L,A,G),L), ((4,L,A,G),L), ((4,L,A,G),H), ((5,H,A,G)  )], 100%, 11.3`
  - Expected sum of rewards = 11.3

Certain case: policies

- *Const High* policies:
  - trajectories, probabilities, rewards:

    ```
    [((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
     ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )], 100%, 9.7
    ```
  - Expected sum of rewards = 9.7.
- *Const Low* policies:
  - trajectories, probabilities, rewards

    ```
    [((0,H,U,G),L), ((0,L,U,G),L), ((0,L,U,G),L), ((0,L,U,G),L), ((0,L,A,G),L),
     ((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G),L), ((0,L,A,G), )], 100%, 10.5
    ```
  - Expected sum of rewards = 10.5
- Optimal policies:
  - trajectories, probabilities, rewards

    ```
    [((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),L),
     ((4,L,A,G),L), ((4,L,A,G),L), ((4,L,A,G),L), ((4,L,A,G),H), ((5,H,A,G)  )], 100%, 11.3
    ```
  - Expected sum of rewards = 11.3
- Optimal policies dictate postponing emission reductions until effective technologies for reducing emissions become available!

Uncertainty on implementability: basic facts

Uncertainty on implementability: basic facts

- $pS2 = pA1 = 0$, $pS1 = pA2 = 1$ but ...

Uncertainty on implementability: basic facts

- $pS2 = pA1 = 0$, $pS1 = pA2 = 1$ but ...
- ... $pLL = pHH = 0.9$ and $pLH = pHL = 0.7$
- Effective technologies still become available after 4 steps and the state of the world turns bad after 6 steps at high emissions but ...

Uncertainty on implementability: basic facts

- $pS2 = pA1 = 0$, $pS1 = pA2 = 1$ but ...
- ... $pLL = pHH = 0.9$ and $pLH = pHL = 0.7$
- Effective technologies still become available after 4 steps and the state of the world turns bad after 6 steps at high emissions but ...
- ... a policy (optimal or not) now entails $2^9 = 512$ possible trajectories.

Uncertainty on implementability: policies

- *Const High* policies:

Uncertainty on implementability: policies

- *Const High* policies:
  - trajectories, probabilities, rewards

```
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )], 38.7%,  9.7
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((8,L,A,B)  )],  4.3%,  9.6
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((4,L,A,G),H), ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B), )],  3.3%, 10.1
...
```

Uncertainty on implementability: policies

- *Const High* policies:
  - trajectories, probabilities, rewards

```
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )], 38.7%,  9.7
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((8,L,A,B)  )],  4.3%,  9.6
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((4,L,A,G),H), ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B), )],  3.3%, 10.1
...
```

  - Expected sum of rewards = 9.904.

Uncertainty on implementability: policies

- *Const High* policies:
  - trajectories, probabilities, rewards

```
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )], 38.7%,  9.7
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((8,L,A,B)  )],  4.3%,  9.6
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((4,L,A,G),H), ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B), )],  3.3%, 10.1
...
```

  - Expected sum of rewards $= 9.904$.
- Optimal policies:

Uncertainty on implementability: policies

- *Const High* policies:
  - trajectories, probabilities, rewards

```
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )], 38.7%,  9.7
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((8,L,A,B)  )],  4.3%,  9.6
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((4,L,A,G),H), ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B), )],  3.3%, 10.1
...
```

  - Expected sum of rewards = 9.904.

- Optimal policies:
  - trajectories, probabilities, rewards

```
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),L), ((2,L,U,G),L), ((2,L,A,G),L),
 ((2,L,A,G),L), ((2,L,A,G),H), ((3,H,A,G),H), ((4,H,A,G),H), ((5,H,A,G), )], 23.4%, 11.2
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),L), ((3,H,U,G),L), ((3,L,A,G),L),
 ((3,L,A,G),L), ((3,L,A,G),L), ((3,L,A,G),H), ((4,H,A,G),H), ((5,H,A,G), )],  7.8%, 11.3
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),L), ((2,L,U,G),L), ((2,L,A,G),L),
 ((2,L,A,G),L), ((2,L,A,G),H), ((2,L,A,G),H), ((3,H,A,G),H), ((4,H,A,G), )],  7.8%, 11.1
...
```

Uncertainty on implementability: policies

- *Const High* policies:
  - trajectories, probabilities, rewards

```
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )], 38.7%,  9.7
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((8,L,A,B)  )],  4.3%,  9.6
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((4,L,A,G),H), ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B), )],  3.3%, 10.1
...
```

  - Expected sum of rewards $= 9.904$.

- Optimal policies:
  - trajectories, probabilities, rewards

```
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),L), ((2,L,U,G),L), ((2,L,A,G),L),
 ((2,L,A,G),L), ((2,L,A,G),H), ((3,H,A,G),H), ((4,H,A,G),H), ((5,H,A,G), )], 23.4%, 11.2
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),L), ((3,H,U,G),L), ((3,L,A,G),L),
 ((3,L,A,G),L), ((3,L,A,G),L), ((3,L,A,G),H), ((4,H,A,G),H), ((5,H,A,G), )],  7.8%, 11.3
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),L), ((2,L,U,G),L), ((2,L,A,G),L),
 ((2,L,A,G),L), ((2,L,A,G),H), ((2,L,A,G),H), ((3,H,A,G),H), ((4,H,A,G), )],  7.8%, 11.1
...
```

  - Expected sum of rewards $= 11.085$.

Uncertainty on implementability: policies

- *Const High* policies:
  - trajectories, probabilities, rewards

```
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )], 38.7%,  9.7
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((8,L,A,B)  )],  4.3%,  9.6
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),H), ((4,H,A,G),H),
 ((4,L,A,G),H), ((5,H,A,G),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B), )],  3.3%, 10.1
...
```

  - Expected sum of rewards = 9.904.
- Optimal policies:
  - trajectories, probabilities, rewards

```
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),L), ((2,L,U,G),L), ((2,L,A,G),L),
 ((2,L,A,G),L), ((2,L,A,G),H), ((3,H,A,G),H), ((4,H,A,G),H), ((5,H,A,G), )], 23.4%, 11.2
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),L), ((3,H,U,G),L), ((3,L,A,G),L),
 ((3,L,A,G),L), ((3,L,A,G),L), ((3,L,A,G),H), ((4,H,A,G),H), ((5,H,A,G), )],  7.8%, 11.3
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),L), ((2,L,U,G),L), ((2,L,A,G),L),
 ((2,L,A,G),L), ((2,L,A,G),H), ((2,L,A,G),H), ((3,H,A,G),H), ((4,H,A,G), )],  7.8%, 11.1
...
```

  - Expected sum of rewards = 11.085.
- Uncertainty about the implementability of decisions dictates earlier emission reductions!

More uncertainties

More uncertainties

- What happens to optimal policies if we account for more uncertainties in the decision problem?

More uncertainties

- What happens to optimal policies if we account for more uncertainties in the decision problem?
- We want to estimate the impacts of:

More uncertainties

- What happens to optimal policies if we account for more uncertainties in the decision problem?
- We want to estimate the impacts of:
  - Uncertainty on the availability of efficient technologies:

More uncertainties

- What happens to optimal policies if we account for more uncertainties in the decision problem?
- We want to estimate the impacts of:
  - Uncertainty on the availability of efficient technologies:
    - There is a small probability that technologies become available before 4 steps and a small probability that technologies do not become available available even after 4 steps!

More uncertainties

- What happens to optimal policies if we account for more uncertainties in the decision problem?
- We want to estimate the impacts of:
  - Uncertainty on the availability of efficient technologies:
    - There is a small probability that technologies become available before 4 steps and a small probability that technologies do not become available available even after 4 steps!
  - Uncertainty on the consequences of exceeding the critical cumulated emission threshold crE:

More uncertainties

- What happens to optimal policies if we account for more uncertainties in the decision problem?
- We want to estimate the impacts of:
  - Uncertainty on the availability of efficient technologies:
    - There is a small probability that technologies become available before 4 steps and a small probability that technologies do not become available available even after 4 steps!
  - Uncertainty on the consequences of exceeding the critical cumulated emission threshold crE:
    - There is a small probability that the world turns bad before 6 high emission steps and a small probability that the world doesn't turns bad even after crE has been exceeded!

Uncertainty on the availability of technologies

Uncertainty on the availability of technologies

- *pLL*, *pHH*, *pLH*, *pHL*, *pS1* and *pS2* as before but ...

Uncertainty on the availability of technologies

- *pLL*, *pHH*, *pLH*, *pHL*, *pS1* and *pS2* as before but ...
- ... $pA1 = 0.1$ and $pA2 = 0.9$ instead of 0 and 1.

Uncertainty on the availability of technologies

- *pLL*, *pHH*, *pLH*, *pHL*, *pS1* and *pS2* as before but ...
- ... $pA1 = 0.1$ and $pA2 = 0.9$ instead of 0 and 1.
- $2^n * (n+1) = 5120$ possible trajectories for a policy sequence for $n = 9$ steps!

Uncertainty on the availability of technologies

- *pLL*, *pHH*, *pLH*, *pHL*, *pS1* and *pS2* as before but ...
- ... $pA1 = 0.1$ and $pA2 = 0.9$ instead of 0 and 1.
- $2^n * (n + 1) = 5120$ possible trajectories for a policy sequence for $n = 9$ steps!
- Optimal policies entail the same most likely trajectories. The expected sum of rewards is almost the same!

Uncertainty on the consequences of exceeding *crE*

Uncertainty on the consequences of exceeding *crE*

- *pLL*, *pHH*, *pLH*, *pHL*, *pA1* and *pA2* as before but ...

Uncertainty on the consequences of exceeding *crE*

- *pLL*, *pHH*, *pLH*, *pHL*, *pA1* and *pA2* as before but ...
- ... $pS1 = 0.9$ and $pS2 = 0.1$ instead of 1 and 0.

Uncertainty on the consequences of exceeding *crE*

- *pLL*, *pHH*, *pLH*, *pHL*, *pA1* and *pA2* as before but ...
- ... $pS1 = 0.9$ and $pS2 = 0.1$ instead of 1 and 0.
- 51200 possible trajectories for a 9-steps policy sequence!

Uncertainty on the consequences of exceeding *crE*

- *pLL*, *pHH*, *pLH*, *pHL*, *pA1* and *pA2* as before but ...
- ... $pS1 = 0.9$ and $pS2 = 0.1$ instead of 1 and 0.
- 51200 possible trajectories for a 9-steps policy sequence!
- For *Const High* policies the most likely trajectory is unchanged but ...

Uncertainty on the consequences of exceeding *crE*

- Optimal policies look now quite different:

Uncertainty on the consequences of exceeding *crE*

- Optimal policies look now quite different:
  - trajectories, probabilities, rewards

```
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),L), ((3,L,A,G),L),
 ((3,L,A,G),L), ((3,L,A,G),L), ((3,L,A,G),H), ((4,H,A,G),H), ((5,H,A,G),)],  5.9%, 11.3

[((0,H,U,G),H), ((1,H,U,B),H), ((2,H,U,B),H), ((3,H,U,B),H), ((4,H,A,B),H),
 ((5,H,A,B),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )],  2.5%,  7.2

[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,B),H), ((3,H,U,B),H), ((4,H,A,B),H),
 ((5,H,A,B),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )],  2.3%,  7.7
...
```

Uncertainty on the consequences of exceeding *crE*

- Optimal policies look now quite different:
  - trajectories, probabilities, rewards

```
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),L), ((3,L,A,G),L),
 ((3,L,A,G),L), ((3,L,A,G),L), ((3,L,A,G),H), ((4,H,A,G),H), ((5,H,A,G),)],  5.9%, 11.3

[((0,H,U,G),H), ((1,H,U,B),H), ((2,H,U,B),H), ((3,H,U,B),H), ((4,H,A,B),H),
 ((5,H,A,B),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )],  2.5%,  7.2

[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,B),H), ((3,H,U,B),H), ((4,H,A,B),H),
 ((5,H,A,B),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )],  2.3%,  7.7
...
```

  - Expected sum of rewards = 9.543

Uncertainty on the consequences of exceeding *crE*

- Optimal policies look now quite different:
  - trajectories, probabilities, rewards

```
[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,G),H), ((3,H,U,G),L), ((3,L,A,G),L),
 ((3,L,A,G),L), ((3,L,A,G),L), ((3,L,A,G),H), ((4,H,A,G),H), ((5,H,A,G),)],  5.9%, 11.3

[((0,H,U,G),H), ((1,H,U,B),H), ((2,H,U,B),H), ((3,H,U,B),H), ((4,H,A,B),H),
 ((5,H,A,B),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )],  2.5%,  7.2

[((0,H,U,G),H), ((1,H,U,G),H), ((2,H,U,B),H), ((3,H,U,B),H), ((4,H,A,B),H),
 ((5,H,A,B),H), ((6,H,A,B),H), ((7,H,A,B),H), ((8,H,A,B),H), ((9,H,A,B), )],  2.3%,  7.7
...
```

  - Expected sum of rewards = 9.543

- Uncertainty about the consequences of exceeding *crE* make earlier emission reductions sub-optimal!

Wrap-up

Wrap-up

- Perhaps not surprisingly, more uncertainty about the implementability of decisions dictates earlier emission reductions. But . . .

Wrap-up

- Perhaps not surprisingly, more uncertainty about the implementability of decisions dictates earlier emission reductions. But ...
- ... more uncertainty about the implications of exceeding critical thresholds make earlier reductions sub-optimal!

Wrap-up

- Perhaps not surprisingly, more uncertainty about the implementability of decisions dictates earlier emission reductions. But . . .
- . . . more uncertainty about the implications of exceeding critical thresholds make earlier reductions sub-optimal!
- Is this what you would have expected?

Wrap-up

- Perhaps not surprisingly, more uncertainty about the implementability of decisions dictates earlier emission reductions. But ...

- ... more uncertainty about the implications of exceeding critical thresholds make earlier reductions sub-optimal!

- Is this what you would have expected? Why?

Wrap-up

- Perhaps not surprisingly, more uncertainty about the implementability of decisions dictates earlier emission reductions. But . . .
- . . . more uncertainty about the implications of exceeding critical thresholds make earlier reductions sub-optimal!
- Is this what you would have expected? Why?
- The results are rigorous: optimality of "optimal" policies is machine-checked!

Thanks for your attention!

- Mail: botta@pik-potsdam.de
- Phone: 288-2657
- Office: A56, 3.26 (Sonne)

- These slides: `https://gitlab.pik-potsdam.de/botta/IdrisLibs/tree/master/lectures/2018-12-03.PIK.Global_change_management_week`