Temporal Logic

Michael Mrozek
Andy Spencer

CSSE490

November 3, 2008

Purpose

The variable foo at some time equals 1, and later equals 2

- $(foo = 1) \vee (foo = 2)$
- $(foo = 1) \wedge (foo = 2)$
- $\mathcal{E} t_1 \mathcal{E} t_2 (t_1 \leq t_2 \wedge ValueAt(foo, t_1) = 1 \wedge ValueAt(foo, t_2) = 2)$
- $\Diamond(foo = 1 \wedge \Diamond(foo = 2))$

Modal Logic

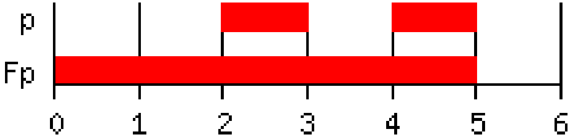
- Formal logic including *modalities*
- Operators for **necessarily** and **possibly**
- Can be written in terms of each other (they form a "**dual pair**")
  - $\Diamond P \leftrightarrow \neg \Box \neg P$
  - $\Box P \leftrightarrow \neg \Diamond \neg P$

Temporal Logic

- A form of modal logic
  - *possibly* $\rightarrow$ *eventually/in the future*
  - *necessarily* $\rightarrow$ *always/globally*
- Propositional logic $+ \Diamond + \Box =$ Propositional temporal logic
- Linear Temporal Logic
- Computational Tree Logic
  - $\mathcal{E}\Box Hungry()$
- Probabilistic CTL

Future ($\Diamond$)

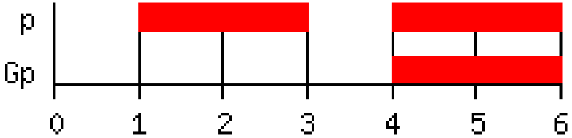
$\Diamond\psi$: ψ must be true sometime in the future



If the user clicks the print button, eventually the file will be printed
$\forall\Box(\mathit{PrintButtonPressed}() \rightarrow \Diamond \mathit{Printed}())$

Globally ($\Box$)

$\Box\psi$: ψ must be true at all times in the future

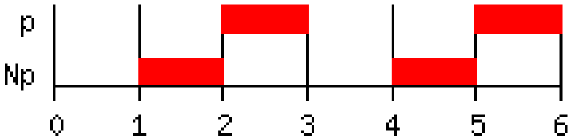


The system will always work and never crash

$\forall\Box(SystemWorking() \wedge \neg SystemCrashed())$

Next ($\bigcirc$)

$\bigcirc\psi$: ψ must be true in the immediate future

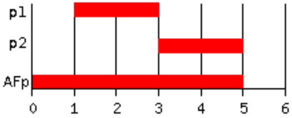


I'm about to eat

$\bigcirc$*Eating*

All ($\mathcal{A}$) and Exists ($\mathcal{E}$)

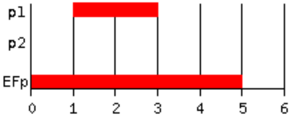
$\mathcal{A}\psi$: ψ must be true in all possible futures



No matter what I will be hungry at some point

$\mathcal{A}\Diamond Hungry()$

$\mathcal{E}\psi$: ψ must be true in at least one possible future

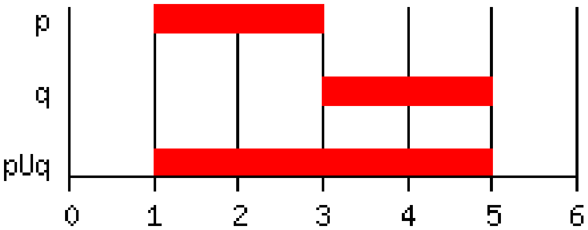


There's a possibility I'll be hungry at some point

$\mathcal{E}\Diamond Hungry()$

Until ($\mathcal{U}$)

$\phi \mathcal{U} \psi$: ψ is true at some point. ϕ is true at least until that time

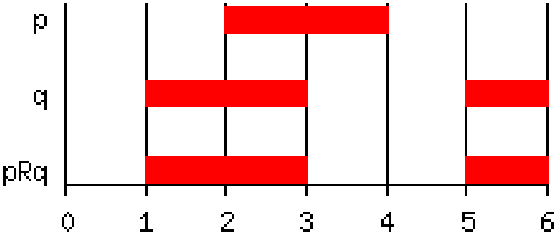


I'll be studying until class starts

$Studying() \mathcal{U} ClassStarts()$

Release ($\mathcal{R}$)

$\phi \mathcal{R} \psi$: ϕ is true until ψ is true, if ever



I'll be studying until class starts

$Studying() \mathcal{R} ClassStarts()$

Laws of Inference

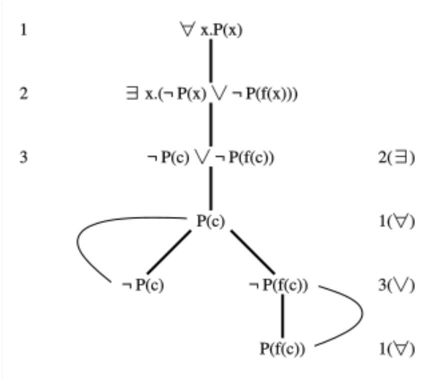
- All predicate logic inference rules
- Distribution of $\Box$ over $\rightarrow$: $\Box(A \rightarrow B) \rightarrow (\Box A \rightarrow \Box B)$
- Distribution of $\bigcirc$ over $\rightarrow$: $\bigcirc(A \rightarrow B) \rightarrow (\bigcirc A \rightarrow \bigcirc B)$
- Expansion of $\Box$: $\Box A \rightarrow (A \wedge \bigcirc A \wedge \bigcirc\Box A)$
- Induction: $\Box(A \rightarrow \bigcirc A) \rightarrow (A \rightarrow \Box A)$
- Linearity: $\bigcirc A \leftrightarrow \neg \bigcirc \neg A$
- Transitivity: $\Box\Box A \leftrightarrow \Box A$
- Distribution: $\bigcirc(A \wedge B) \leftrightarrow (\bigcirc A \wedge \bigcirc B)$
- Distribution: $\Box(A \wedge B) \rightarrow (\Box A \wedge \Box B)$
- Contraction: $A \wedge \bigcirc\Box A \rightarrow \Box A$
- Distribution: $(\Box A \wedge \Box B) \rightarrow \Box(A \wedge B)$
- Exchange: $\Box \bigcirc A \leftrightarrow \bigcirc\Box A$

Soundness/Completeness

Syntactic Consequence: $A \vdash B$
Semantic Consequence: $A \models B$

Soundness: No false positives. $\vdash \rightarrow \models$
Completeness: No false negatives. $\models \rightarrow \vdash$
$\models \Box(A \rightarrow \bigcirc A) \rightarrow (A \rightarrow \Box A)$
$s \models \Box(A \rightarrow \bigcirc A) \wedge A \wedge \neg\Box A$
$s \models \neg\Box A \rightarrow s' \models \neg A$
$s, s_1, s_2, ..., s'$
$s \models A \wedge (A \rightarrow \bigcirc A) \rightarrow s_1 \models A$
$s' \models A \wedge s' \models \neg A \rightarrow \bot$

Soundness/Completeness



Strengths and Weaknesses

Strengths

- Easier to represent common situations
- Easy to learn; few operators
- Works well with existing techniques

Weaknesses

- Not strictly necessary
- Can't represent potentials

Applications

- Natural Language
- Artificial Intelligence
  - Frame problem
  - Event calculus
- Program Specification/Verification

Statement	Progress axioms
`li: v := expression`	$\vdash l_i \rightarrow \Diamond l_{i+1}$

`li: if B then`
`lt: S1`
`else`
`lf: S2`
	$\vdash (l_i \wedge \Box B) \rightarrow \Diamond l_t$
	$\vdash (l_i \wedge \Box \neg B) \rightarrow \Diamond l_f$

`li: while B do`
`lt: S1;`
`lf: S2`
	$\vdash (l_i \wedge \Box B) \rightarrow \Diamond l_t$
	$\vdash (l_i \wedge \Box \neg B) \rightarrow \Diamond l_f$

Applications

- Natural Language
- Artificial Intelligence
  - Frame problem
  - Event calculus
- Program Specification/Verification

Statement	Progress axioms
`li: v := expression`	$\vdash l_i \rightarrow \Diamond l_{i+1}$
`li: if B then`	$\vdash (l_i \wedge \Box B) \rightarrow \Diamond l_t$
`lt: S1`	
`else`	$\vdash (l_i \wedge \Box \neg B) \rightarrow \Diamond l_f$
`lf: S2`	
`li: while B do`	$\vdash (l_i \wedge \Box B) \rightarrow \Diamond l_t$
`lt: S1;`	
`lf: S2`	$\vdash (l_i \wedge \Box \neg B) \rightarrow \Diamond l_f$