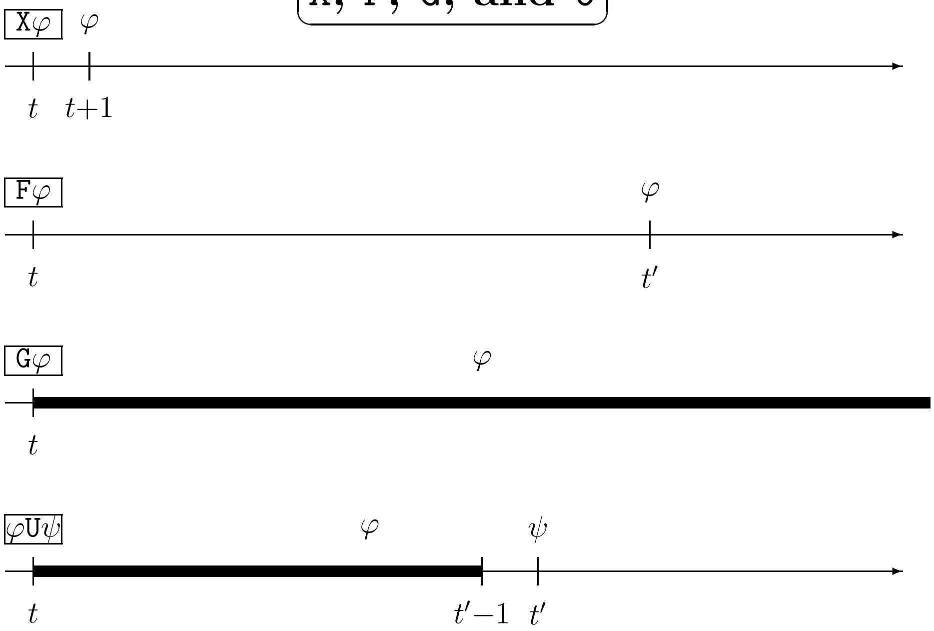
X, F, G, and U



Semantics of LTL

Let $\mathcal{A} = (\{1, \ldots, \ell\}, <, P_1, P_2, \ldots, P_n)$ and $t \in \{1, \ldots, \ell\}$.
Define $\mathcal{A}, t \models \varphi$ for $\varphi \in \mathsf{LTL}[p_1, \ldots, p_2]$ inductively.

- $\mathcal{A}, t \models \top$; $\mathcal{A}, t \not\models \bot$; $\mathcal{A}, t \models p_i$ iff $t \in P_i$
- $\mathcal{A}, t \models \varphi \wedge \psi$ iff ...
- $\mathcal{A}, t \models \mathtt{X}\varphi$ iff $t < \ell$ and $\mathcal{A}, t{+}1 \models \varphi$
- $\mathcal{A}, t \models \mathtt{F}\varphi$ iff $\mathcal{A}, t' \models \varphi$ for some $t' \geq t$
- $\mathcal{A}, t \models \mathtt{G}\varphi$ iff $\mathcal{A}, t' \models \varphi$ for all $t' \geq t$
- $\mathcal{A}, t \models \varphi\mathtt{U}\psi$ iff there is some $t' \geq t$ s.t. $\mathcal{A}, t' \models \psi$ and $\mathcal{A}, t'' \models \varphi$ for all $t \leq t'' < t'$

In the previous slide we just abbreviated the model relation for the propositional connectives, as they are unchanged. Recall the slide from an earlier lecture.

Model Relation for Propositional Logic

Writing $\underline{\mathrm{a}} \models \varphi$ for $\varphi[\underline{\mathrm{a}}] = 1$ we obtain the following.

- $\underline{\mathrm{a}} \models \top$ always holds and $\underline{\mathrm{a}} \models \bot$ never holds
- $\underline{\mathrm{a}} \models \neg\varphi$ holds iff $\underline{\mathrm{a}} \models \varphi$ does *not* hold
- $\underline{\mathrm{a}} \models \varphi \wedge \psi$ holds if $\underline{\mathrm{a}} \models \varphi$ *and* $\underline{\mathrm{a}} \models \psi$ both hold.
  $\underline{\mathrm{a}} \models \varphi \vee \psi$ holds if $\underline{\mathrm{a}} \models \varphi$ holds *or* $\underline{\mathrm{a}} \models \psi$ holds.

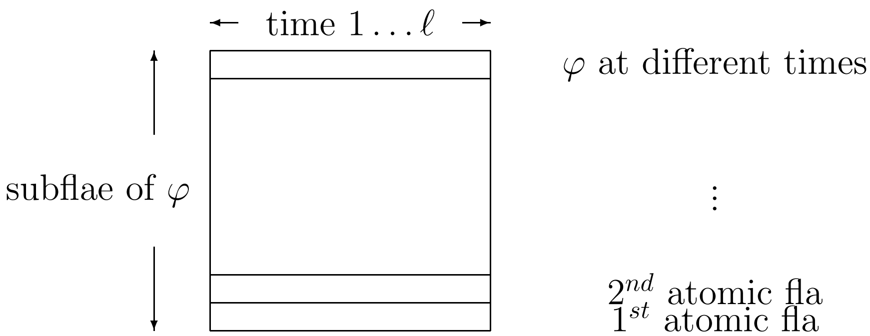
Recall the model-checking problem, which, for LTL reads as follows.

> **Given:** $\varphi \in \mathsf{LTL}[p_1, \ldots, p_2]$ and $\mathcal{A} = (\{1, \ldots, \ell\}, <, P_1, P_2, \ldots, P_n)$ and $1 \leq t \leq \ell$
>
> **Question:** Does $\mathcal{A}, t \models \varphi$?

For propositional logic, we observed that the truth of $\underline{a} \models \varphi$ only depends on the truth of $\underline{a} \models \varphi'$ for subformulae φ' of φ. Here we can do a similar observation: the truth of $\mathcal{A}, t \models \varphi$ only depends on that of $\mathcal{A}, t' \models \varphi'$ for φ' a subformulae of φ.

Hence, a similar dynamic programming approach will work.

> **Answer:** Polynomial in $|\varphi|$ and $|\mathcal{A}|$.



We can do even better than that! Looking again at the definition of the $\mathcal{A}, t \models \varphi$ relation, we note that we can reformulate it in such a way, that truth only depends on the truth of proper subformulae at the same time, or a (non-proper) subformula at the next point in time.

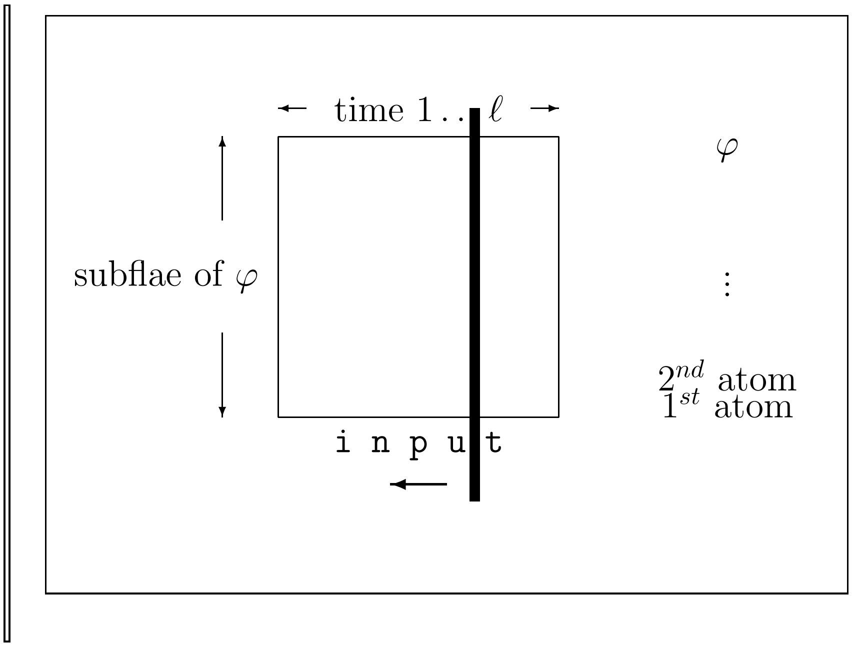
LTL Model Checking

t		t		$t{+}1$
$\mathcal{A}, t \models \mathtt{X}\varphi$	iff			$\mathcal{A}, t{+}1 \models \varphi$
$\mathcal{A}, t \models \mathtt{F}\varphi$	iff	$\mathcal{A}, t \models \varphi$	or	$\mathcal{A}, t{+}1 \models \mathtt{F}\varphi$
$\mathcal{A}, t \models \mathtt{G}\varphi$	iff	$\mathcal{A}, t \models \varphi$	and	$\mathcal{A}, t{+}1 \models \mathtt{G}\varphi$
				in case $t + 1 \leq \ell$
$\mathcal{A}, t \models \varphi\mathtt{U}\psi$	iff	$\mathcal{A}, t \models \psi$	or	
		$\mathcal{A}, t \models \varphi$	and	$\mathcal{A}, t{+}1 \models \varphi\mathtt{U}\psi$

On the one hand, the above proof now gives the better time complexity, since the amount of lookup for any particular filed is now constant.

Time $\mathcal{O}(|\mathcal{A}| \cdot |\varphi|)$

Moreover, the amount of information we have to store only depends on the size of the formula. We can just sweep through the word/trace and update the information about all subformulae.



Let's see this form of model checking in an example.

Example for LTL Model Checking

Formula $\mathtt{G}(y \rightarrow y\mathtt{U}r)$. *That this* $\mathtt{G}(\neg y \vee y\mathtt{U}r)$.

$\mathcal{A}$	r	1	1	1	0	0	0	0	0	1
	y	0	0	1	0	0	0	1	1	0
	g	0	0	0	1	1	1	0	0	0
$\mathtt{G}(y \rightarrow y\mathtt{U}r)$										
$y \rightarrow y\mathtt{U}r$										
$y\mathtt{U}r$										
$\neg y$										

Note that, when working our way from the end of the trace to its beginning, we could, at any point, forget any future columns after the earliest one we filled.

This is precisely the characteristic behaviour of an automaton walking (backwards) over the word.

LTL and Automata

For φ construct DFA $\mathcal{M}_\varphi$ with $L(\mathcal{M}_\varphi) = \{w | w^{-1} \models \varphi\}$.

States: sets of sub-formulae of φ.
indicating with formulae hold at a given position

Transitions: Given

- previous state $\{\psi \mid \mathcal{A}, t{+}1 \models \psi\}$
- t'th letter of w, i.e., local properties of $\mathcal{A}$ at time t

determine $\{\psi \mid \mathcal{A}, t \models \psi\}$ by the rules seen.

This description by an automata has several important consequence. First of all, we get decidability of the satisfiability problem. This was not a priory clear. As opposed to propositional logic, we didn't have an a priory bound on the candidate structures (as words can have arbitrary length!).

> **Theorem.** Satisfiability for an LTL-formula φ over finite words can be decided in *space* polynomial $|\varphi|$.
>
> **Proof.** Decide the emptiness-problem for the automaton associated with φ.
> *(Since a state is described by $|\varphi|$ many bits, we are in non-deterministic space $\mathcal{O}(|\varphi|)$, which is in deterministic space $\mathcal{O}(|\varphi|^2)$.)*

Moreover, the fact that the number of states of the automaton has "only" $2^{|\varphi|}$ states gives us a bound on the size of the smallest structure satisfying an LTL property.

Corollary. If an LTL-formula φ can be satisfied by a finite structure, then it can also be satisfied by a structure of size at most $2^{|\varphi|}$.

Also note that from a DFA for w^{-1} we can get directly an NFA for w (and therefore also an DFA at the price of an exponential increase in the number of states).

All LTL-definable properties are regular.