Verification of liveness properties
Programs and $\omega$-executions

- Recall: a **full execution** of a program is an execution that cannot be extended (either infinite or ending at a configuration without successors).
- We consider programs that may have $\omega$-executions.
- We assume w.l.o.g. that every full execution of the program is infinite (see next slide).
- Therefore: **full executions** $= \omega$-executions
Handling finite full executions

We artificially ensure that every full execution is infinite by adding a self-loop to every state without successors.
Verifying a program

• **Goal**: automatically check if some $\omega$-execution violates a property.

• **Safety property**: “nothing bad happens”
  – No configuration satisfies $x = 1$.
  – No configuration is a deadlock.
  – Along an execution the value of $x$ cannot decrease.

• **Liveness property**: “something good eventually happens”
  – Eventually $x$ has value $1$.
  – Every message sent during the execution is eventually received.
Safety and liveness: more precisely

• A finite execution \( w \) is **bad** for a given property if every potential \( \omega \)-execution of the form \( w \ w' \) violates the property.

• A property is a safety property if every \( \omega \)-execution that violates the property has a bad prefix.
  (Intuitively: after finite time we can already say that the property does not hold)

• A property is a liveness property if some \( \omega \)-execution that violates the property has no bad prefix.
  (We can only tell that the property is a violation `after seeing the complete \( \omega \)-execution`).
Approach to automatic verification

• Represent the set of $\omega$-executions of the program as a NBA. (The system NBA).

• Represent the set of possible $\omega$-executions that violate the property as a NBA (or an $\omega$-regular expression). (The property NBA).

• Check emptiness of the intersection of the two NBAs.
Problem: Fairness

- We may want to exclude some $\omega$-executions because they are “unfair”.
- Example: finite waiting property in Lamport’s mutex algorithm.
Lamport’s algorithm
Asynchronous product
Finite waiting property

- **Finite waiting**: If a process is trying to access the critical section, it eventually will.

- Formalization: Let $NC_i, T_i, C_i$ be atomic propositions mapped to the sets of configurations where process $i$ is in the non-critical section, trying to access it, and in the critical section, respectively. The full executions that violate finite waiting for process $i$ are

  $$\Sigma^* T_i (\Sigma \setminus C_i)^\omega$$

- Observe: all states of the system NBA are final, and so we can intersect NBAs using the algorithm for NFAs.
Finite waiting property

• The finite waiting property does not hold because of

\[ [0,0, nc_0, nc_1] \ [1,0, t_0, nc_1] \ [1,1, t_0, t_1]^\omega \]

• Is this a real problem of the algorithm? No! We have not specified correctly.

• **Fairness assumption**: both processes execute infinitely many actions.
  (Usually a weaker assumption is used: if a process can execute actions infinitely often, it executes infinitely many actions.)

• Reformulation: in every fair \( \omega \)-execution, if a process is trying to access the critical section, it will eventually access it.
Finite waiting property

• The violations of the property under fairness are the intersection of $\Sigma^* T_i (\Sigma \setminus C_i)^\omega$ and the $\omega$-executions in which both processes make a move infinitely often.

• Problem: how do we represent this condition as an $\omega$-regular language?

• Solution: enrich the alphabet of the NBA Letter: pair $(c, i)$ where $c$ is a configuration and $i$ is the index of the process making the move.
Finite waiting property

• Denote by $M_0$ and $M_1$ the set of letters with index 0 and 1, respectively.

• The possible $\omega$ executions where both processes move infinitely often is given by

$$\left( (M_0 + M_1)^* M_0 M_1 \right)^\omega$$

• Finite waiting holds under fairness for process 0 but not for process 1 because of

$$\left( [0,0,nc_0,nc_1][0,1,nc_0,t_1][1,1,t_0,t_1][1,1,t_0,q_1] [1,0,t_0,q'_1][1,0,c_0,q'_1][0,0,nc_0,q'_1] \right)^\omega$$
Temporal logic

- Writing property NBAs requires training in automata theory
- We search for a more intuitive (but still formal) description language: Temporal Logic.
  - Temporal logic extends propositional logic with temporal operators like always and eventually.
  - Linear Temporal Logic (LTL) is a temporal logic interpreted over linear structures.
Linear Temporal Logic (LTL)

• We are given:
  – A set \( AP \) of atomic propositions (names for basic properties)
  – A valuation assigning to each atomic proposition a set of configurations (intended meaning: the set of configurations that satisfy the property).
Example

```plaintext
while x = 1 do
  if y = 1 then
    x ← 0
  y ← 1 - x
end
```

- $AP : a_{t_1}, a_{t_2}, ... , a_{t_5}, x=0, x=1, y=0, y=1$
- $V(a_{t_i}) = \{[l, x, y] \in C \mid l = i\}$ for every $i \in \{1, ..., 5\}$
- $V(x=0) = \{[l, x, y] \in C \mid x = 0\}$
A computation is an infinite sequence of subsets of $AP$.

Examples for $AP = \{p, q\}$

$$\emptyset^\omega \quad (\{p\}\{p, q\})^\omega \quad \{p\} \{p, q\} \emptyset \emptyset \{p\}^\omega$$

We map every possible execution to a computation by mapping each configuration to the set of atomic propositions it satisfies.

A computation is executable if some execution maps to it.
Example

\[ e_1 = [1,0,0] [5,0,0]^{\omega} \]

\[ e_2 = ( [1,1,0] [2,1,0] [4,1,0] )^{\omega} \]

\[ e_3 = [1,0,1][5,0,1]^{\omega} \]

\[ e_4 = [1,1,1][2,1,1][3,1,1][4,0,1][1,0,1][5,0,1]^{\omega} \]
From executions to computations

\[ e_1 = [1,0,0] [5,0,0]^\omega \]

\[ e_2 = ( [1,1,0] [2,1,0] [4,1,0] )^\omega \]

\[ \sigma_1 = \{ \text{at1, x}=0, y=0 \} \{ \text{at5, x}=0, y=0 \}^\omega \]

\[ \sigma_2 = ( \{ \text{at1, x}=0, y=0 \} \{ \text{at2, x}=1, y=0 \} \{ \text{at4, x}=1, y=0 \} )^\omega \]
• Given: set $AP$ of atomic propositions, valuation assigning to each atomic proposition a set configurations.

• The formulas of LTL are given by the syntax:

$$\varphi ::= \text{true} \mid p \mid \neg \varphi_1 \mid \varphi_1 \land \varphi_2 \mid X\varphi_1 \mid \varphi_1 U \varphi_2$$

where $p \in AP$
Semantics of LTL

• Formulas are interpreted on computations (executable or not).
• The satisfaction relation $\sigma \models \varphi$ is given by:

  $\sigma \models \text{true}$
  $\sigma \models p$ iff $p \in \sigma(0)$
  $\sigma \models \neg \varphi$ iff $\text{not } \sigma \models \varphi$
  $\sigma \models \varphi_1 \land \varphi_2$ iff $\sigma \models \varphi_1$ and $\sigma \models \varphi_2$
  $\sigma \models X\varphi$ iff $\sigma^1 \models \varphi$
  $\sigma \models \varphi_1 U \varphi_2$ iff there is $k \geq 0$ s.t. $\sigma^k \models \varphi_2$ and $\sigma^i \models \varphi_1$ for all $0 \leq i \leq k$
Abbreviations

• The boolean abbreviations *false*, *∨*, →, ↔ etc. are defined as usual.

• $F\varphi := \text{true} \cup \varphi$ (eventually $\varphi$).

According to the semantics:

\[ \sigma \models F\varphi \text{ iff there is } k \geq 0 \text{ s.t. } \sigma^k \models \varphi \]

• $G\varphi := \neg F\neg\varphi$ (always $\varphi$ or globally $\varphi$).

According to the semantics:

\[ \sigma \models G\varphi \text{ iff } \sigma^k \models \varphi \text{ for every } k \geq 0 \]
Examples of formulas

• $AP: \ at_1, \ at_2, \ldots, \ at_5$, $x=0, \ x=1, \ y=0, \ y=1$

  $V(at_i) = \{[\ell, x, y] \in C | \ell = i\}$ for every $i \in \{1, \ldots, 5\}$

  $V(x=0) = \{[\ell, x, y] \in C | x=0\}$

• $\varphi_0 = x=1 \land X \ y=1 \land X \ X \ at3$

• $\varphi_1 = F \ x=0$

• $\varphi_2 = x=0 \ U \ at5$

• $\varphi_3 = y=1 \land F( x=0 \land at5 ) \land \neg( F( y=0 \land X \ y=1 ) )$
Lamport’s algorithm

- \( AP = \{ NC_0, T_0, C_0, NC_1, T_1, C_1, M_0, M_1 \} \)

Valuation as expected.

- Mutual exclusion:

- Naïve finite waiting:

- Finite waiting with fairness:
Bounded overtaking:

\[ G \left( T_0 \rightarrow \left( \neg C_1 \cup \left( C_1 \cup \left( \neg C_1 \cup C_0 \right) \right) \right) \right) \]

Whenever \( T_0 \) holds, the computation continues with a (possibly empty) interval at which \( \neg C_1 \) holds, followed by a (possibly empty) interval at which \( C_1 \) holds, followed by a point at which \( C_0 \) holds.
Getting used to LTL

• Express in natural language $FGp$, $GFp$
• Are these pairs of formulas equivalent?

<table>
<thead>
<tr>
<th>$FFp$</th>
<th>$Fp$</th>
<th>$GGp$</th>
<th>$Gp$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$FGp$</td>
<td>$GFp$</td>
<td>$FGFp$</td>
<td>$GFp$</td>
</tr>
<tr>
<td>$p \cup q$</td>
<td>$p \cup (p \land q)$</td>
<td>$p \cup q$</td>
<td>$p \land (p \cup q)$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$Fp$</th>
<th>$p \lor XFp$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Gp$</td>
<td>$p \lor XGp$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$p \cup q$</th>
<th>$p \lor X (p \cup q)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q \lor X (p \cup q)$</td>
<td></td>
</tr>
<tr>
<td>$q \lor (p \land X (p \cup q))$</td>
<td></td>
</tr>
<tr>
<td>$q \land (p \lor X (p \cup q))$</td>
<td></td>
</tr>
</tbody>
</table>
From formulas to NBAs

- Given: set $AP$ of atomic propositions
- Language $L(\varphi)$ of a formula $\varphi$ : set of computations satisfying $\varphi$.
- Examples for $AP = \{p, q\}$
  - $L(Fp) =$ computations $s_1s_2s_3 \ldots$ such that $p \in s_i$ for some $i \geq 1$
  - $L(G(p \land q)) = \{\{p, q\}\omega\}$
- $L(\varphi)$ is an $\omega$-language over the alphabet $2^{AP}$
- For $AP = \{p, q\}$ we get $2^{AP} = \{\emptyset, \{p\}, \{q\}, \{p, q\}\}$
NBAs for some formulas

\[ AP = \{p, q\} \]

- \( Fp \)
- \( Gp \)
- \( p \cup q \)
- \( GFp \)
We present an algorithm that takes a formula $\varphi$ over a fixed set $AP$ of atomic propositions as input and returns a NGA $A_\varphi$ such that $L(A_\varphi) = L(\varphi)$. 
Closure of a formula

- Define $\text{neg}(\psi) = \begin{cases} \psi & \text{if } \varphi = \neg \psi \\ \neg \varphi & \text{otherwise} \end{cases}$
- The closure $\text{cl}(\varphi)$ of $\varphi$ is the set containing $\psi$ and $\text{neg}(\psi)$ for every subformula $\psi$ of $\varphi$
- Example:

$$\text{cl}(p \lor \neg q) = \{p, \neg p, \neg q, q, p \lor \neg q, \neg (p \lor \neg q)\}$$
• The **satisfaction sequence** of a computation $s_0s_1s_2 \ldots$ with respect to $\varphi$ is the sequence $\alpha_0\alpha_1\alpha_2 \ldots$ where $\alpha_i$ contains the formulas of $cl(\varphi)$ satisfied by $s_is_{i+1}s_{i+2} \ldots$
Satisfaction sequence

• The satisfaction sequence of a computation $s_0s_1s_2 \ldots$ with respect to $\varphi$ is the sequence $\alpha_0\alpha_1\alpha_2 \ldots$ where $\alpha_i$ contains the formulas of $cl(\varphi)$ satisfied by $s_is_{i+1}s_{i+2} \ldots$

• The satisfaction sequence of $\{p\}^\omega$ w.r.t. $p U q$ is:
Satisfaction sequence

• The satisfaction sequence of a computation $s_0s_1s_2 \ldots$ with respect to $\varphi$ is the sequence $\alpha_0\alpha_1\alpha_2 \ldots$ where $\alpha_i$ contains the formulas of $cl(\varphi)$ satisfied by $s_is_{i+1}s_{i+2} \ldots$

• The satisfaction sequence of $\{p\}^\omega$ w.r.t. $p U q$ is:
  $$\{p, \neg q, \neg (p U q)\}^\omega$$
• The satisfaction sequence of a computation \( s_0 s_1 s_2 \ldots \) with respect to \( \varphi \) is the sequence \( \alpha_0 \alpha_1 \alpha_2 \ldots \) where \( \alpha_i \) contains the formulas of \( \text{cl}(\varphi) \) satisfied by \( s_i s_{i+1} s_{i+2} \ldots \).

• The satisfaction sequence of \( \{p\}^\omega \) w.r.t. \( p \cup q \) is:

\[
\{p, \neg q, \neg (p \cup q)\}^\omega
\]

• The satisfaction sequence of \( (\{p\}\{q\})^\omega \) w.r.t. \( p \cup q \) is:
Satisfaction sequence

• The satisfaction sequence of a computation $s_0s_1s_2 \ldots$ with respect to $\varphi$ is the sequence $\alpha_0\alpha_1\alpha_2 \ldots$ where $\alpha_i$ contains the formulas of $\text{cl}(\varphi)$ satisfied by $s_is_{i+1}s_{i+2} \ldots$

• The satisfaction sequence of $\{p\}^\omega$ w.r.t. $p U q$ is:

$$\{p, \neg q, \neg (p U q)\}^\omega$$

• The satisfaction sequence of $\{p\}\{q\}^\omega$ w.r.t. $p U q$ is:

$$\left(\{p, \neg q, p U q\} \{\neg p, q, p U q\}\right)^\omega$$
Atoms

• Intuition: an atom is a maximal set of formulas of \( cl(\varphi) \) that “could be simultaneously true by looking only at \( \neg \) and \( \land \)”
Atoms

• Intuition: an atom is a maximal set of formulas of $cl(\varphi)$ that “could be simultaneously true by looking only at $\neg$ and $\land$”

• A set $\alpha \subseteq cl(\varphi)$ is an atom if it satisfies the following two conditions:
  – For every $\psi \in cl(\varphi)$, exactly one of $\psi$ and $\neg(\psi)$ belong to $\alpha$
  – For every $\psi_1 \land \psi_2 \in cl(\varphi)$, $\psi_1 \land \psi_2 \in \alpha$ iff $\psi_1 \in \alpha$ and $\psi_2 \in \alpha$
Atoms

• Intuition: an atom is a maximal set of formulas of \( cl(\varphi) \) that “could be simultaneously true by looking only at \( \neg \) and \( \land \)”

• A set \( \alpha \subseteq cl(\varphi) \) is an atom if it satisfies the following two conditions:
  – For every \( \psi \in cl(\varphi) \), exactly one of \( \psi \) and \( \neg(\psi) \) belong to \( \alpha \)
  – For every \( \psi_1 \land \psi_2 \in cl(\varphi), \psi_1 \land \psi_2 \in \alpha \) iff \( \psi_1 \in \alpha \) and \( \psi_2 \in \alpha \)

• Examples of atoms for \( \varphi = (\neg p \land q) U Fp \) :
Atoms

• Intuition: an atom is a maximal set of formulas of $cl(\varphi)$ that “could be simultaneously true by looking only at ¬ and ∧”

• A set $\alpha \subseteq cl(\varphi)$ is an atom if it satisfies the following two conditions:
  – For every $\psi \in cl(\varphi)$, exactly one of $\psi$ and neg($\psi$) belong to $\alpha$
  – For every $\psi_1 \land \psi_2 \in cl(\varphi)$, $\psi_1 \land \psi_2 \in \alpha$ iff $\psi_1 \in \alpha$ and $\psi_2 \in \alpha$

• Examples of atoms for $\varphi = \neg (p \land q) U Fp$:
  $\{\neg p, \neg q, \neg (p \land q), Fp, \varphi\} \quad \{p, q, (p \land q), \neg Fp, \neg \varphi\}$
Atoms

• Intuition: an atom is a maximal set of formulas of $cl(\varphi)$ that “could be simultaneously true by looking only at $\neg$ and $\land$”

• A set $\alpha \subseteq cl(\varphi)$ is an atom if it satisfies the following two conditions:
  – For every $\psi \in cl(\varphi)$, exactly one of $\psi$ and $\text{neg}(\psi)$ belong to $\alpha$
  – For every $\psi_1 \land \psi_2 \in cl(\varphi)$, $\psi_1 \land \psi_2 \in \alpha$ iff $\psi_1 \in \alpha$ and $\psi_2 \in \alpha$

• Examples of atoms for $\varphi = \neg(p \land q) \cup Fp$:
  \begin{align*}
  \{\neg p, \neg q, \neg (p \land q), Fp, \varphi\} & \quad \{p, q, (p \land q), \neg Fp, \neg \varphi\}
  \end{align*}

• Examples of non-atoms for $\varphi = \neg(p \land q) \cup Fp$:
• Intuition: an atom is a maximal set of formulas of $cl(\varphi)$ that “could be simultaneously true by looking only at $\neg$ and $\land$”

• A set $\alpha \subseteq cl(\varphi)$ is an atom if it satisfies the following two conditions:
  – For every $\psi \in cl(\varphi)$, exactly one of $\psi$ and $\text{neg}(\psi)$ belong to $\alpha$
  – For every $\psi_1 \land \psi_2 \in cl(\varphi)$, $\psi_1 \land \psi_2 \in \alpha$ iff $\psi_1 \in \alpha$ and $\psi_2 \in \alpha$

• Examples of atoms for $\varphi = \neg(p \land q) U Fp$:
  $\{\neg p, \neg q, (p \land q), Fp, \varphi\}$  $\{p, q, (p \land q), \neg Fp, \neg \varphi\}$

• Examples of non-atoms for $\varphi = \neg(p \land q) U Fp$:
  $\{p, q, p \land q, Fp\}$  $\{p \land q, Fp, \varphi\}$
Hintikka sequences

• A pre-Hinttika sequence for $\varphi$ is a sequence $\alpha_0\alpha_1\alpha_2 \ldots$ of subsets of $cl(\varphi)$ satisfying the following conditions for every $i \geq 0$:
  
  – For every $X\psi \in cl(\varphi)$:
    
    $X\psi \in \alpha_i$ iff $\psi \in \alpha_{i+1}$
  
  – For every $\psi_1 U \psi_2 \in cl(\varphi)$:
    
    $\psi_1 U \psi_2 \in \alpha_i$ iff $\psi_2 \in \alpha_i$ or $\psi_1 \in \alpha_i$ and $\psi_1 U \psi_2 \in \alpha_{i+1}$
Hintikka sequences

- A pre-Hinttika sequence for \( \varphi \) is a sequence \( \alpha_0 \alpha_1 \alpha_2 \ldots \) of subsets of \( cl(\varphi) \) satisfying the following conditions for every \( i \geq 0 \):
  - For every \( X\psi \in cl(\varphi) \):
    \( X\psi \in \alpha_i \) iff \( \psi \in \alpha_{i+1} \)
  - For every \( \psi_1 U \psi_2 \in cl(\varphi) \):
    \( \psi_1 U \psi_2 \in \alpha_i \) iff \( \psi_2 \in \alpha_i \) or \( \psi_1 \in \alpha_i \) and \( \psi_1 U \psi_2 \in \alpha_{i+1} \)
- A pre-Hinttika sequence is a Hinttika sequence if it also satisfies for every \( i \geq 0 \):
  - For every \( \psi_1 U \psi_2 \in cl(\varphi) \): if \( \psi_1 U \psi_2 \in \alpha_i \) then there exists \( j \geq i \) such that \( \psi_2 \in \alpha_j \)
Hintikka sequences: An example

• Let $\varphi = \neg (p \land q) \cup (r \land s)$. Which of the following are pre-Hintikka and Hintikka sequences?
Hintikka sequences: An example

• Let $\varphi = \neg(p \land q) \cup (r \land s)$. Which of the following are pre-Hintikka and Hintikka sequences?

1. $\{p, \neg q, r, s, \varphi\}$
Hintikka sequences: An example

• Let $\varphi = \neg (p \land q) \cup (r \land s)$. Which of the following are pre-Hintikka and Hintikka sequences?

1. $\{p, \neg q, r, s, \varphi\}^\omega$
2. $\{\neg p, r, \neg \varphi\}^\omega$
Hintikka sequences: An example

• Let $\varphi = \neg(p \land q) U (r \land s)$. Which of the following are pre-Hintikka and Hintikka sequences?
  1. $\{p, \neg q, r, s, \varphi\}^\omega$
  2. $\{\neg p, r, \neg \varphi\}^\omega$
  3. $\{\neg p, q, \neg r, r \land s, \neg \varphi\}^\omega$
Hintikka sequences: An example

• Let $\varphi = \neg(p \land q) U (r \land s)$ . Which of the following are pre-Hintikka and Hintikka sequences?

1. $\{p, \neg q, r, s, \varphi\}^\omega$
2. $\{\neg p, r, \neg \varphi\}^\omega$
3. $\{\neg p, q, \neg r, r \land s, \neg \varphi\}^\omega$
4. $\{p, q, p \land q, r, s, r \land s, \neg \varphi\}$
Hintikka sequences: An example

• Let $\varphi = \neg(p \land q) \cup (r \land s)$. Which of the following are pre-Hintikka and Hintikka sequences?

1. $\{p, \neg q, r, s, \varphi\}^\omega$
2. $\{\neg p, r, \neg \varphi\}^\omega$
3. $\{\neg p, q, \neg r, r \land s, \neg \varphi\}^\omega$
4. $\{p, q, p \land q, r, s, r \land s, \neg \varphi\}$
5. $\{p, \neg q, \neg(p \land q), \neg r, s, \neg(r \land s), \varphi\}^\omega$
Hintikka sequences: An example

- Let $\varphi = \neg(p \land q) \cup (r \land s)$. Which of the following are pre-Hintikka and Hintikka sequences?

1. $\{p, \neg q, r, s, \varphi\}^\omega$
2. $\{\neg p, r, \neg \varphi\}^\omega$
3. $\{\neg p, q, \neg r, r \land s, \neg \varphi\}^\omega$
4. $\{p, q, p \land q, r, s, r \land s, \neg \varphi\}$
5. $\{p, \neg q, \neg (p \land q), \neg r, s, \neg (r \land s), \varphi\}^\omega$
6. $\{p, q, (p \land q), r, s, (r \land s), \varphi\}^\omega$
Main theorem

- **Definition:** A Hintikka sequence $\alpha_0 \alpha_1 \alpha_2 \ldots$ extends a computation $s_0 s_1 s_2 \ldots$ if $s_i \cap cl(\varphi) = \alpha_i \cap AP$ for every $i \geq 0$.

- **Theorem:** Every computation $s_0 s_1 s_2 \ldots$ can be extended to a unique Hintikka sequence, and this extension is equal to the satisfaction sequence.
Strategy for the NFA of a formula

• Let $\sigma$ be a computation over $AP$. 
Strategy for the NFA of a formula

• Let $\sigma$ be a computation over $AP$.
• We have: $\sigma \models \varphi$
  iff $\varphi$ belongs to the first set of the satisfaction sequence for $\sigma$
  iff $\varphi$ belongs to the first set of the Hintikka sequence for $\sigma$
Strategy for the NFA of a formula

• Let $\sigma$ be a computation over $AP$.
• We have: $\sigma \models \varphi$

  iff $\varphi$ belongs to the first set of the satisfaction sequence for $\sigma$

  iff $\varphi$ belongs to the first set of the Hintikka sequence for $\sigma$

• Strategy: design the NGA so that for every $\sigma$
  – The runs on $\sigma$ correspond to the pre-Hintikka sequences $\alpha_0 \alpha_1 \alpha_2 \ldots$ that extend $\sigma$ and satisfy $\varphi \in \alpha_0$
  – A run is accepting iff its corresponding pre-Hintikka sequence is also a Hintikka sequence.
The NGA $A_\varphi$
The NGA $A_\varphi$

- Alphabet: $2^{AP}$
The NGA $A_\varphi$

- **Alphabet:** $2^{AP}$
- **States:** atoms of $\varphi$. 
The NGA $A_\varphi$

- **Alphabet:** $2^{AP}$
- **States:** atoms of $\varphi$.
- **Initial states:** atoms containing $\varphi$. 
The NGA $A_\varphi$

- **Alphabet:** $2^{AP}$
- **States:** atoms of $\varphi$.
- **Initial states:** atoms containing $\varphi$.
- **Transitions:** triples $\alpha \rightarrow^s \beta$ such that $\alpha \cap \{p, \neg p \mid p \in AP\} = s$ and $\alpha, \beta$ satisfies the conditions of a pre-Hintikka sequence.
The NGA $A_\varphi$

- **Alphabet:** $2^{AP}$
- **States:** atoms of $\varphi$.
- **Initial states:** atoms containing $\varphi$.
- **Transitions:** triples $\alpha \rightarrow \beta$ such that $\alpha \cap AP = s$ and $\alpha, \beta$ satisfies the conditions of a pre-Hintikka sequence.
- **Sets of accepting states:** A set $F_{\psi_1 U \psi_2}$ for every until-subformula $\psi_1 U \psi_2$ of $\varphi$.

$F_{\psi_1 U \psi_2}$ contains the atoms $\alpha$ such that $\psi_1 U \psi_2 \notin \alpha$ or $\psi_2 \in \alpha$. 
Example: The NGA $A_p U q$

(Labels of transitions omitted. The label of a transition from atom $\alpha$ is the set $\{p \in AP \mid p \in \alpha\}$. There is only one set of accepting states.)
Some observations

• All transitions leaving a state carry the same label.
• For every computation $s_0s_1s_2 \ldots$ satisfying $\varphi$ there is a unique accepting run $\alpha_0 \rightarrow s_0 \alpha_1 \rightarrow s_1 \alpha_2 \rightarrow s_2 \rightarrow \ldots$, namely the one such that $\alpha_0 \alpha_1 \alpha_2 \ldots$ is the satisfaction sequence for $s_0s_1s_2 \ldots$.
• The sets of computations accepted from each initial state are pairwise disjoint.
• The number of states is bounded by $2^{\lvert \varphi \rvert}$. 