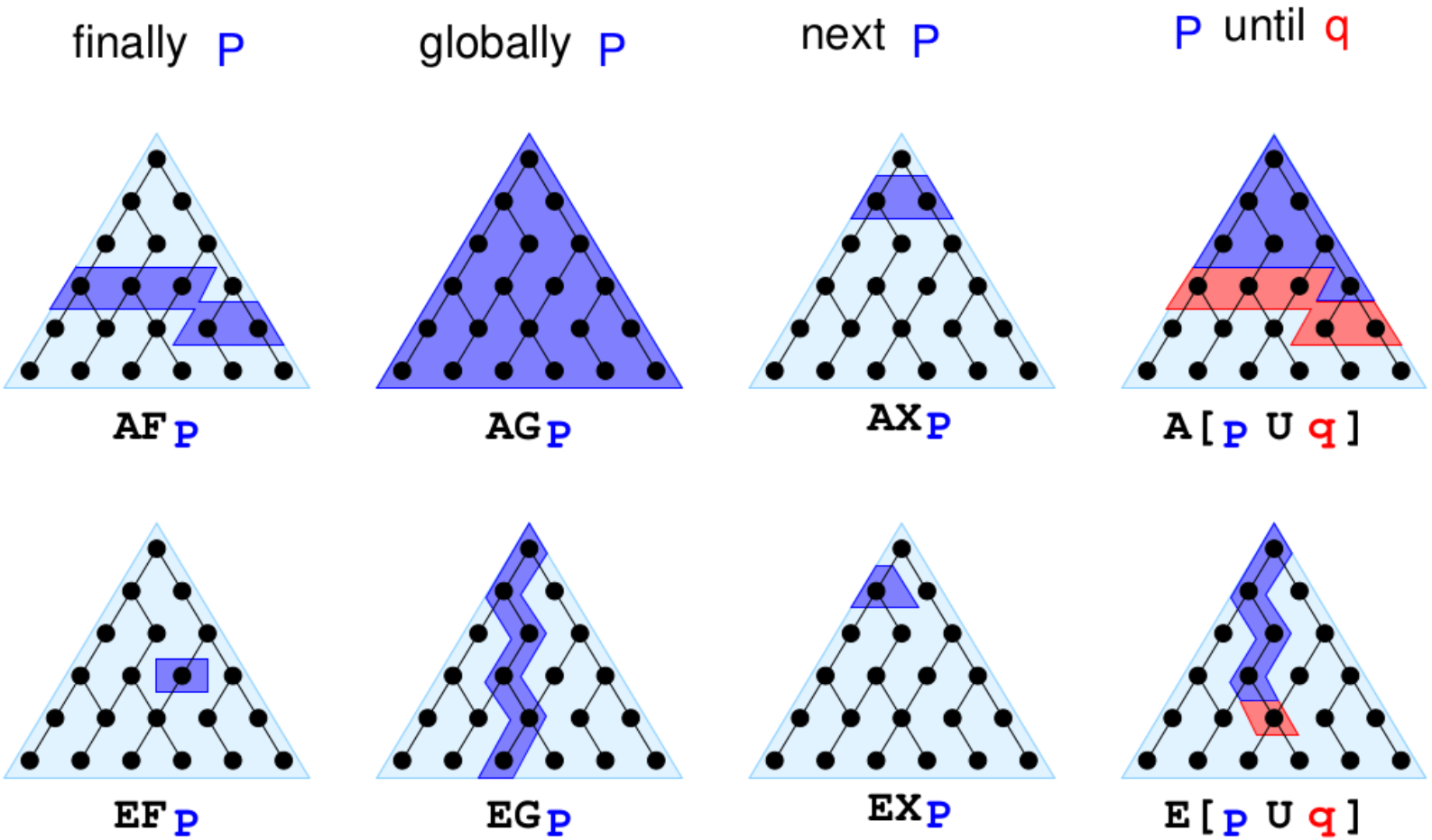
CTL intuitive recall



(C) Alessandro Artale

Known picture drawback - the runs should form trees!

CTL examples

Express in CTL:

- A state where a is true, but b is not, is reachable.
- Whenever system receives a request *Req* then it generates an acknowledgement *Ack* eventually.
- Whenever system receives a request *Req* then it is possible that it will generate an acknowledgement *Ack* eventually.
- In every run there are infinitely many b.

CTL examples

Express in CTL:

- All the paths lead to Rome.
- All the time if I have not died yet, then I have a chance to survive one more day.
- All the time if I get robbed then I can react by defending myself or not defending myself.

CTL examples

Read CTL formula:

- $AG[$ error $\implies E($repair U operational$)$ $]$
- $AG[$ error $\implies AX\ A($!error W operational$)$ $]$
- $AG[\ EF($restart$)$ $]$
- $AG[\ EX($restart$)$ $]$
- $A[\ p\ U\ A(q\ U\ r)$ $]$

How to read:

- AX, EX - necessarily next, possibly next
- AF - necessarily in the future (or Inevitably)
- EF - possibly in the future (or Possibly)
- AG - globally (or Always)
- $AG(\phi \implies \psi)$ - Whenever ϕ then ψ.
- EG - possibly henceforth
- AU, EU - necessarily until, possibly until

CTL properties

- $\neg AG\varphi \equiv EF\neg\varphi$
- $\neg EG\varphi \equiv AF\neg\varphi$
- $\neg EX\varphi \equiv AX\neg\varphi$

- discuss $\neg Gp$ in LTL

- $EX(\varphi \vee \psi) \equiv EX\varphi \vee EX\psi$
- $EX(\varphi \wedge \psi) \not\equiv EX\varphi \wedge EX\psi$
- $AX(\varphi \vee \psi) \not\equiv AX\varphi \vee AX\psi$
- $AX(\varphi \wedge \psi) \equiv AX\varphi \wedge AX\psi$

CTL vs. LTL examples

Express in LTL: $EF[\ a \wedge \neg b]$

Compare the following formulae:

- $AG[\ EF$ restart $]$ vs. $G[\ \neg(G\neg$ restart$)\ ]$
- $AG[\ p \implies AF\ q\ ]$ vs. $G(\ p \implies Fq\ )$
- $AF[\ AG\ p\ ]$ vs. $FG\ p$
- $AG[\ AF\ p\ ]$ vs. $GF\ p$
- $AF[\ AX\ p\ ]$ vs. $FX\ p$

Express in CTL: $(GF\ p \wedge GF\ q) \implies \psi$