1. TAD Guesteros

generos juego

igualdad observacional

$$(\forall g, g' : \text{guesteros}) \left(g =_{\text{obs}} g' \iff \begin{pmatrix} \text{g} =_{\text{obs}} \text{g'} \iff (\text{Reinos(g)} =_{\text{obs}} \text{Reinos(g')} \wedge_l \\ \text{Alianzas(g)} =_{\text{obs}} \text{Alianzas(g')} \wedge_l \\ \text{Espias(g)} =_{\text{obs}} \text{Espias(g')} \wedge_l \\ \text{Batallas(g)} =_{\text{obs}} \text{Batallas(g'))} \end{pmatrix} \right)$$

observadores básicos

Reinos : Guesteros G $\longrightarrow$ conj(Reinos)
Alianzas : Guesteros G $\times$ Reino R $\longrightarrow$ conj(Reinos) {R ε conj(reinos)}
Espias : Guesteros G $\times$ Reino R $\longrightarrow$ conj(Reinos) {R ε conj(reinos)}
Batallas : Guesteros G $\longrightarrow$ secu(batallas)

generadores

Iniciar : conj(Reinos)CR $\longrightarrow$ Guesteros
Atacar : Reino R $\times$ Reino R' $\times$ Guesteros G $\longrightarrow$ Guesteros
{R' no pertenece a Alianzas(G,R) $\wedge$ R,R' ε Reinos(G)}
Aliarse : Reino R $\times$ Reino R' $\times$ Guesteros G $\longrightarrow$ Guesteros
{R' no pertenece a Alianzas(G,R) $\wedge$ R,R' ε Reinos(G)}
Espiar : Reino R $\times$ Reino R' $\times$ Guesteros G $\longrightarrow$ Guesteros
{R' no pertenece a Alianzas(G,R) $\wedge$ R,R' ε Reinos(G)}

otras operaciones

cantSoldadosTotalesAtacante : Reino R $\times$ Guesteros G $\longrightarrow$ Nat
{R ε Reinos(G)}
cantSoldadosDefensor : Reino R' $\times$ Reinos R $\times$ Guesteros G $\longrightarrow$ Nat
{R,R' ε Reinos(G)}
calcularPerdidas : Reino R' $\times$ Reinos R $\times$ Guesteros G $\longrightarrow$ Nat
{R,R' ε Reinos(G)}
ocupar : Reino R $\times$ Reinos R' $\times$ Guesteros G $\longrightarrow$ conj(Reinos)
{R,R' ε Reinos(G)}
destruirAlianza : Reino R $\times$ Reinos R' $\times$ conj(Reinos) $\times$ Guesteros G $\longrightarrow$ conj(Reinos)
{R,R' ε Reinos(G)}
batallasMasSangrienta : Guesteros G $\longrightarrow$ Nat
batallasMasSangrientaAux : Nat $\times$ Secu(Nat) $\longrightarrow$ Nat
losers : Guesteros G $\longrightarrow$ conj(Reinos)
losersAuxiliar : Reino R $\times$ conj(Reinos) $\longrightarrow$ conj(Reinos)
{R ε Reinos(G)}
losersAux : conj(Reinos)) $\longrightarrow$ conj(Reinos)

axiomas

Reinos(iniciar(CR)) $\equiv$ CR
Reinos(atacar(R,R'.G)) $\equiv$ **if** cantSoldadosTotalesAtacante?(R,G) $>$ cantSoldadosDefensor(R',R,G) **then**
THEN
ag(<π_1(R'),π_2(R')-porc(,π_2(R),30)), Reinos(G)-R))
else
ag(<π_1(R),π_2(R)-porc(,π_2(R),30)), Reinos(G)-R)
fi
Reinos(aliarse(R,R',G)) $\equiv$ Reinos(G)
Reinos(espiar(R,R',G)) $\equiv$ Reinos(G)
Alianzas(iniciar(CR), R) $\equiv$ $\emptyset$

Alianzas(atacar(R,R',G), R") ≡ **if** cantSoldadosTotalesAtacante?(R,G) > cantSoldadosDefensor(R',R,G) **then**
IF (π_2(R)-porc(π_2(R),10)) > calcularPerdidos(R',R,G)) THEN
ocupar(R,R'.G)
FI
else
IF ((π_2(R)-porc(π_2(R),10)) > calcularPerdidos(R',R,G)) THEN
ocupar(R',R,G)
fi

Alianzas(espias(G,R), R) ≡ alianzas(G,R

Alianzas(aliarse(R,R',G), R") ≡ ag(R',alianzas(G,R)

Espias(iniciar(CR),R) ≡ $\emptyset$

Espias(atacar(R,R',G),R) ≡ espias(G,R)

Espias(aliarse(R,R',G),R) ≡ espias(G,R)

Espias(aliarse(R,R',G),R") ≡ espias(G,R)

Batallas(iniciar(cj)) ≡ <>

Batallas(atacar(R,R',G)) ≡ **if** cantSoldadosAtacante(R,G) > cantSoldadosDefensor(R,R',G) **then**
Batallas(G) ∘ porc(π_2(R)-porc(π_2(R'),30))
else
Batallas(G) ∘ porc(π_2(R)-porc(π_2(R),30))
fi

Batallas(aliarse(R,R',G)) ≡ Batallas(G)

Batallas(espiar(R,R',G)) ≡ Batallas(G)

cantSoldadosTotalesAtacante(R,R',G ≡ cantSoldadosTotalAux(R,alianzas(G,R'),alianza(G,R),G

cantSoldadosTotalesAux(R,AR',AR,G ≡ **if** $\emptyset?(AR)$ **then**
$\pi_2(c)$
else
IF dameuno(AR) ε AR' THEN
cantSoldadosAux(R,AR',SinUno(AR),G) ELSE
π_2(dameuno(AR)+cantSoldadosAux(R,AR',SinUno(AR),G)
fi
FI

Tambien puede ser que R sea aliado de algunos espiado por R'

cantSoldadosDefensor(R',R,G ≡ **if** R ε espias(G,R') **then**
2*cantSoldadosTotalAux(R',Alianza(R,G), Alianzas(R',G'),G
else
cantSoldadosTotalAux(R',Alianza(R,G), Alianzas(R',G),G)
fi

calcularPerdidos(R,G) ≡ porc(π_2(R)-porc(π_2(R'),30)

ocupar(R,R',G) ≡ destruirAlianzas(R,R',Alianzas(R',G),G

destruirAlianzas(R,R',CR,G ≡ **if** $\emptyset$?(CR) **then**
ag(R,$\emptyset$
else
destruirAlianzas(R,R',sinUNo(CR),G)
fi

BatallasMasSangrienta(G ≡ BatallasMasSangrientaAix(prim(Batalla(G)),fin(Batalla(G))

BatallasMasSangrientaAux(n.nx ≡ **if** vacia?(ns) **then**
n
else
IF (n $\geq$ prim(nx)) THEN BatallaMasSangrientaAux(N,fin(nx) ELSE BatallaMasSangrientaAux(prin(nx), fin(nx))
fi FI

losers(G) ≡ losersAux(Reinos(G))

losersAuxiliar(R,CR) $\equiv$ **if** $\emptyset?(CR)$ **then**
True
else
IF cantSoldadosDefensor(R,dameUno(CR),G) < cantSoldadosAtacantes(dameUno(CR),R,G) THEN
loser(R, sinUno(CR)) ELSE False
fi
FI

losersAux(CR) $\equiv$ **if** $\emptyset?(CR)$ **then**
$\emptyset$
else
IF losersAuxiliar(dameuno(CR),dameUNO(CR)-Alianzas(dameUno(CR))-CR) THEN
ag(dameUno(CR),loserAux(SinUno(CR))
fi
FI

2) a) Falso. Tomando f(n)=k$\log_2$(n), $\forall k > 1 \longrightarrow 2^{klog_2(n)}$ no pertenece a O(n)

b) Verdadero. Como f y g son funciones naturales, la multiplicacion de dos numeros naturales siempre sera mayor o igual al minimo de ambos numeros. Es decir, supongamos que g(n) (lo mismo para f(n))
(Existe $n_0 \varepsilon$ **N**, k>0 / n_0 $\geq$n $\longrightarrow$ f(n)g(n)$\geq$k(g(n)) pero, como la imagen de ambas funciones son naturales, la multiplicacion de ambas sera mayor a g(n) si tomo k=1 para cualquier n_0

c)El peor caso del algoritmo es aquel donde el primer elemento coincide con 1 y que el n sea mayor, entonces A[pos] no se mueve y ese entra al for interno O(n^2) veces ya que entra al menos n^2 veces como minimo ya que se cumpla la condicion del for una vez y que coincide min y max.

d) Falso, si tomo un n natural menor a A[1], entonces el algortimo no entra en el ciclo y eso es O(1).