Mad Max

Ivan Geffner Salvador Roura

23 de novembre de 2018

1 Regles del Joc

En aquest joc, 4 jugadors tenen control sobre un exèrcit de guerrers i cotxes en un tauler de 60×60 caselles amb 8 ciutats. L'objectiu del joc és "conquerir" el major nombre possible de ciutats.

Els equips s'identifiquen amb nombres de 0 a 3. Inicialment, cada equip té 2 de les ciutats. A cada ronda, el nombre de ciutats de cadascun dels equips s'afegeix a la puntuació corresponent. El guanyador del joc és el jugador amb la puntuació més alta després de l'última ronda.

Al principi del joc, cada equip té 23 unitats: 20 guerrers i 3 cotxes. Els guerrers comencen a l'interior de les ciutats dels seus equips, amb almenys un guerrer a cada ciutat. Els cotxes comencen a les vores del tauler, cadascun en una carretera.

El tauler té sis tipus de caselles: desert (*Desert*), ciutat (*City*), carretera (*Road*), aigua (*Water*), estació de servei (*Station*) i paret (*Wall*). Els guerrers poden passar pel desert, les ciutats i les carreteres. Els cotxes només es poden desplaçar pel desert i les carreteres. Cap casella pot tenir més d'una unitat. Les caselles d'aigua recarreguen guerrers adjacents amb aigua, les estacions de servei recarreguen cotxes adjacents amb combustible, i les parets són obstacles que no es poden travessar, sense cap altra finalitat.

El joc dura 500 rondes, numerades de la 0 a la 499. Les unitats del jugador x només es poden moure a les rondes que siguin congruents amb x mòdul 4. Per exemple, el jugador 1 només es pot moure a les rondes 1, 5, 9, etc. Hi ha una excepció: els cotxes que tenen combustible i que estan ubicats sobre una carretera, poden desplaçar-se independentment de la ronda. A més, cada unitat es pot moure com a molt una vegada per ronda.

Al final de cada ronda, i per cada ciutat, si un equip té estrictament més guerrers dins d'aquesta ciutat que qualsevol altre equip, llavors aquest equip conquereix la ciutat (tret que la ciutat ja fos propietat d'aquest equip, en què la propietat no canvia). Una ciutat és un component connex de caselles de ciutat (horitzontalment i vertical).

Les unitats poden intentar moure's cap a qualsevol casella adjacent horitzontalment, vertical o diagonal, sempre que el tipus de la casella adjacent sigui compatible amb el tipus de la unitat. Si la casella adjacent està buida, la unitat es mou allà. En cas contrari, s'apliquen algunes normes (vegeu a sota). Les unitats no es poden moure fora del tauler.

Inicialment, els guerrers tenen 40 unitats d'aliment i també 40 unitats d'aigua, i els cotxes tenen 100 unitats de combustible. Totes les rondes que se'ls permeti

moure's, els guerrers perden 1 unitat d'aliment i 1 unitat d'aigua, i els cotxes perden 1 unitat de combustible. Un guerrer amb un nivell d'aliment o d'aigua que arriba a 0 mor, i un equip rival triat a l'atzar obté un altre guerrer. Un cotxe sense combustible no "mor", sinó que es torna lent no només en el desert, sinó també a les carreteres. Les unitats mai no poden tenir més aliment, aigua o combustible que la seva quantitat inicial.

Qualsevol guerrer en una casella de ciutat recarrega el seu nivell d'aliment a 40, independentment de l'equip que posseeixi la ciutat. Qualsevol guerrer adjacent a una de les caselles d'aigua recarrega el seu nivell d'aigua a 40. Qualsevol cotxe adjacent a una casella d'estació de servei recarrega el seu nivell de combustible fins a 100. Els guerrers hauran de creuar almenys una carretera per arribar a l'aigua des d'una ciutat (o de tornada). Les estacions de servei sempre estaran ubicades en algun encreuament de carreteres.

Un guerrer pot intentar moure's cap a una casella on hi hagi un altre guerrer o cotxe. De la mateixa manera, un cotxe pot intentar moure's cap a una casella on hi hagi un guerrer o un altre cotxe. Primer expliquem les regles que s'apliquen en el cas habitual que els dos equips implicats siguin diferents.

Un cotxe que es mou a una casella amb un guerrer l'atropella i el mata (el captura). Cada moment, qualsevol unitat morta desapareix del tauler i una nova unitat apareix a la següent ronda, tal com s'explica a baix.

Si un cotxe es mou cap a una casella amb un altre cotxe, els dos cotxes exploten i desapareixen, i es donen dos nous cotxes als altres dos equips.

Un guerrer que es mou a una casella amb un cotxe es suïcida, i es dóna un nou guerrer a l'equip rival.

Els guerrers poden atacar qualsevol guerrer adjacent amb una ordre per moure's a la posició del rival. En general (quan l'atacat té almenys 6 unitats d'aliment i d'aigua), el guerrer atacat perdrà 6 unitats d'aliment i 6 unitats d'aigua. A més, si algun d'aquests nivells arriba a 0, l'atacat serà capturat per l'equip rival. A més, l'atacant es recarregarà amb la meitat de l'aliment i aigua robats (normalment 3 unitats de cadascun), mai superant el límit de 40.

Hi ha una excepció a aquesta regla: si hi ha dos guerrers dins d'una ciutat i un d'ells ataca l'altre, llavors s'utilitza la regla de la cúpula del tro: "Dos homes hi entren, només un home en surt". Després d'una lluita sagnant, un dels guerrers mor, mentre que la salut del guanyador roman sense canvis.

El supervivent d'una lluita de cúpula de tro es decideix de forma aleatòria, amb probabilitats proporcionals als nivells d'aigua dels lluitadors. Per exemple, si un té 30 unitats d'aigua i l'altre 20 unitats, llavors el primer sobreviurà amb probabilitat $3/5$, mentre que l'últim sobreviurà amb probabilitat $2/5$.

Noteu que es permet que les unitats ataquin unitats del mateix equip. Com a regla general, això probablement serà una mala idea, tot i que en alguns casos pot ser útil sacrificar una unitat pròpia per tal de salvar-ne una altra.

Quan l'atacant i l'atacat pertanyen al mateix equip, les regles s'adapten com es descriu a continuació. Si un guerrer mor, el nou guerrer serà donat a un altre equip. Si un cotxe xoca amb un altre cotxe, els dos nous cotxes seran donats a dos equips diferents. Tots aquests nous equips seran escollits aleatòriament de forma uniforme.

Com a resultat d'aquestes regles, el nombre total de guerrers i el nombre total de cotxes roman constant durant tot el joc.

A cada ronda es pot donar més d'una ordre a la mateixa unitat, encara que només se seleccionarà la primera (si n'hi ha). Qualsevol programa que intenti donar més de 1000 ordres durant la mateixa ronda serà avortat.

Els moviments seleccionats de cada ronda s'executaran en ordre aleatori. Per exemple, si un guerrer intenta passar a la posició d'un cotxe amic, i aquest cotxe intenta passar a un lloc buit, llavors si el guerrer es mou primer, se suïcidarà i després el cotxe es mourà. En cas contrari, si el cotxe es mou primer, les dues unitats es mouran i sobreviuran. Un altre exemple: si dos cotxes enemics intenten atropellar un guerrer d'un altre equip, que no es mou, l'equip del primer cotxe en moure's captarà el guerrer, i després els dos cotxes xocaran i explotaran.

Després d'executar tots els moviments seleccionats d'una ronda, les unitats capturades per cada equip reneixen. Sempre que sigui possible, els cotxes seran col·locats en carreteres a les vores del tauler, i els guerrers a caselles de desert també a les vores, sempre a una distància raonable d'altres unitats. En els casos rars en què això no es pugui aconseguir, els cotxes seran col·locats en carreteres no a les vores, i els guerrers en caselles de desert no a les vores, també a una distància raonable d'altres unitats. En situacions encara més excepcionals, els cotxes es col·locaran en qualsevol casella legal i els guerrers també en qualsevol casella legal, però mai dins d'una ciutat.

Si us calen nombres (pseudo) aleatoris, heu d'usar els dos mètodes proporcionats pel joc: `random(l, u)`, que retorna un enter aleatori en `[l..u]`, i (menys freqüentment) `random_permutation(n)`, que retorna un `vector<int>` amb una permutació aleatòria de `[0..n-1]`.

Noteu que les direccions vàlides són `Bottom`, `BR`, `Right`, `RT`, `Top`, `TL`, `Left`, `LB` and `None`, corresponents a enters de 0 a 8. Aquesta definició circular es pot usar per simplificar la implementació del vostre jugador. Vegeu el jugador `Demo` com a exemple.

Un joc es defineix per un tauler i el següent conjunt de paràmetres:

- *nb_players*: Nombre d'equips en el joc (4).
- *nb_rounds*: Nombre de rondes que es jugaran (500).
- *nb_cities* : Nombre de ciutats en el tauler (8).
- *nb_warriors*: Nombre inicial de guerrers per jugador (20).
- *nb_cars*: Nombre inicial de cotxes per jugador (3).
- *warriors_health* : Nivell màxim (i inicial) d'aliment i d'aigua de cada guerrer (40).
- *cars_fuel* : Nivell màxim (i inicial) de combustible de cada cotxe (100).

- *damage*: La quantitat de mal que un guerrer infligeix quan ataca (6).
- *rows*: Mida vertical del tauler (60).
- *cols*: Mida horitzontal del tauler (60).

2 Programar el joc

El primer que heu de fer és descarregar el codi font. Inclou un programa de C++ que executa les partides i un visor HTML per veure-les en un format animat raonable. A més, també es proporciona un jugador "Null" i un jugador "Demo" per facilitar la implementació del vostre propi jugador.

2.1 Executar la primera partida

Aquí explicarem com executar el joc sota Linux, però el mateix hauria de funcionar sota Windows, Mac, FreeBSD, OpenSolaris, ... Només cal una versió recent de `g++`, `make`, i un navegador modern com Firefox o Chrome.

1. Obriu la consola i feu `cd` al directori on heu extret el codi font.
2. Executeu

   ```
   make all
   ```

   per construir el joc i tots els jugadors. Noteu que `Makefile` identifica com a jugador qualsevol fitxer `AI*.cc`.
3. Això crea un fitxer executable anomenat `Game`. Aquest executable us permet executar el joc usant una comanda com:

   ```
   ./Game Demo Demo Demo Demo -s 30 -i default.cnf -o default.res
   ```

   Això comença una partida, amb llavor aleatòria 30, de quatre clons del jugador Demo, sobre el tauler definit a `default.cnf`. La sortida d'aquesta partida es redirigeix a `default.res`.
4. Per veure la partida, obriu el fitxer visor `viewer.html` amb el navegador i carregueu el fitxer `default.res`.

Useu

```
./Game --help
```

per veure la llista de paràmetres que podeu usar. És particularment útil

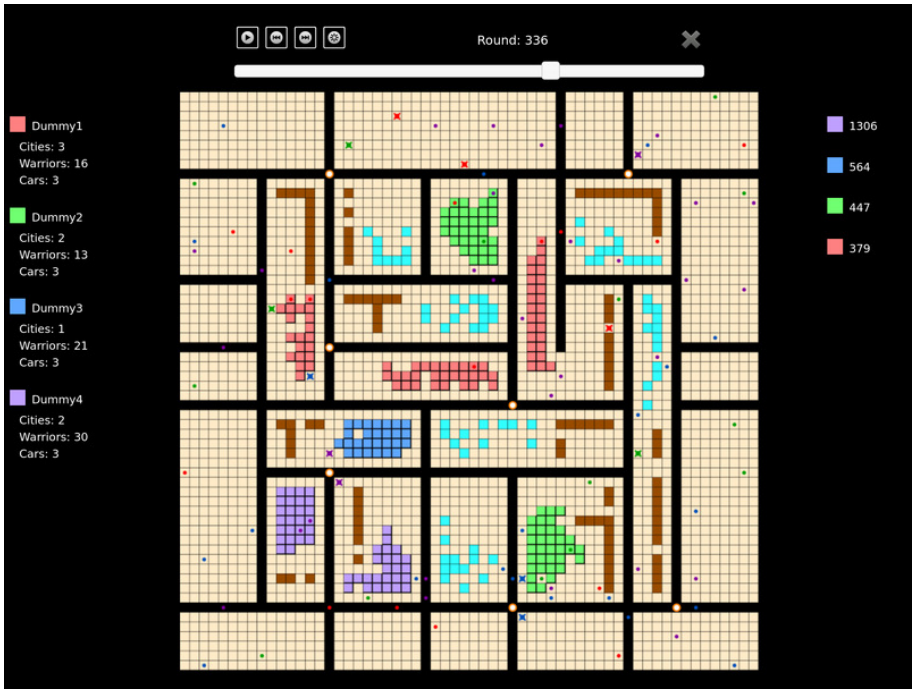
```
./Game --list
```

per veure tots els noms de jugadors reconeguts.

Si cal, recordeu que podeu executar

```
make clean
```

per esborrar tots els fitxers executables i objectes i començar de nou.



2.2 Afegir el vostre jugador

Per crear un nou jugador amb, per exemple, nom `Rockatansky`, copieu `AINull.cc` (un jugador buit que es proporciona com a plantilla) a un nou fitxer `AIRockatansky.cc`. Aleshores, editeu el nou fitxer i canvieu el

#define PLAYER_NAME Null

per

#define PLAYER_NAME Rockatansky

El nom escollit per al vostre jugador ha de ser únic, no ofensiu i de com a molt 12 caràcters de longitud. Aquest nom serà mostrat a la web i durant les partides.

Després, podeu començar implementant el mètode virtual *play* (), heretat de la classe base *Player*. Aquest mètode, que serà cridat a cada ronda, ha de decidir les ordres que doneu a les vostres unitats.

Podeu definir tipus auxiliars, variables i mètodes dins de la vostra classe jugador, però el punt d'entrada del vostre codi serà sempre el mètode *play* ().

De la vostra classe jugador també podeu cridar funcions per accedir a l'estat del joc. Aquestes funcions estan disponibles per al vostre codi usant herència. La documentació sobre les funcions disponibles es pot trobar en un fitxer addicional.

Noteu que no heu d'editar el mètode *factory* () de la vostra classe jugador, ni tampoc la darrera línia que afegeix el vostre jugador a la llista de jugadors disponibles.

2.3 Jugar contra el jugador "Dummy"

Per provar la vostra estratègia contra el "Dummy", us proporcionem fitxers objecte `AIDummy.o` d'aquest jugador. D'aquesta manera no teniu accés al codi font, però encara podeu afegir-lo com a jugador i competir contra ell localment.

Per afegir el jugador "Dummy" a la llista de jugadors registrats, heu d'editar el fitxer `eMakefile` i donar a la variable `DUMMY_OBJ` el valor apropiat. Recordeu que els fitxers objecte contenen instruccions binàries per a màquines específiques, de manera que no podem proporcionar un únic fitxer genèric. Si us cal un fitxer objecte per a la vostra arquitectura, contacteu-nos i intentarem proporcionar-vos-el.

També podeu demanar als vostres amics els seus fitxers objecte i afegir-los al `Makefile` donant valor a la variable `EXTRA_OBJ`.

2.4 Restriccions en l'enviament d'un jugador

Quan creieu que el vostre jugador és prou fort com per entrar a la competició, podeu enviar-lo al Jutge. Com que s'executarà en un entorn segur per evitar trampes, cal aplicar algunes restriccions al vostre codi:

- Tot el vostre codi font ha d'estar en un sol fitxer (com AIRockatansky.cc).
- No podeu usar variables globals (en lloc d'això, useu atributs de la vostra classe).
- Només se us permet usar llibreries estàndard com ara `iostream`, `vector`, `map`, `set`, `queue`, `algorithm`, `cmath`, ... En molts casos, no cal ni tan sols que incloeu la corresponent llibreria.
- No podeu obrir fitxers ni fer altres crides a sistema (threads, forks, ...).

- El vostre temps de CPU i memòria seran limitats (tot i que no ho estaran en el vostre entorn local quan executeu `./Game`).
- El vostre programa no hauria d'escriure a **cout** ni llegir de **cin**. Podeu escriure informació de debug a **cerr**, però recordeu que fer això en en codi que pugeu pot malgastar part del vostre temps limitat de CPU.
- Qualsevol enviament al Jutge ha de ser un intent honest de jugar el joc. Qualsevol intent de fer trampa de qualsevol manera serà severament penalitzat.

3 Consells

- Llegeix les capçaleres de les classes que utilitzaràs. No et preocupis per les parts privades o la implementació.
- Comenceu amb estratègies simples, fàcils de codificar i depurar, ja que això és exactament el que necessiteu al principi.
- Definiu mètodes auxiliars simples (però útils) i *assegureu-vos que funcionin correctament).*
- Abans de competir amb els vostres companys de classe, concentreu-vos en derrotar el "Dummy".
- Conserveu una còpia de les versions anteriors del vostre jugador. Feu que lluiti contra les seves versions anteriors per mesurar les millores.
- Com sempre, compileu i proveu el codi sovint. És *molt* més fàcil traçar un error quan només heu canviat poques línies de codi.
- Useu **cerr** per escriure informació de depuració i afegiu `assert`'s per assegurar-vos que el codi fa el que hauria de fer. Recordeu d'eliminar-los abans de carregar el codi per evitar fer-lo més lent.
- En depurar un jugador, elimineu els **cerr** 's que pugui haver-hi en el codi dels altres jugadors, per veure només els missatges que desitgeu.
- Si l'ús de **cerr** no és suficient per depurar algun codi, apreneu com utilitzar `valgrind`, `gdb` o qualsevol altra eina de depuració.
- Assegureu-vos que el programa sigui prou ràpid. El temps de CPU que se us permet utilitzar és bastant curt.
- Intenteu esbrinar les estratègies d'altres jugadors observant diverses partides. D'aquesta manera, podeu intentar reaccionar contra ells o fins i tot imitar-los o millorar-los en el vostre propi codi.

- No doneu el vostre codi a ningú. Ni tan sols una versió antiga. Ni fins i tot al vostre millor amic. Utilitzem detectors de plagi per comparar els programes, també contra enviaments d'anys anteriors.
- Podeu, però, compartir els fitxers `.o` compilats.
- Podeu enviar noves versions del vostre programa en qualsevol moment.
- No espereu fins a l'últim moment per enviar el jugador. Quan hi ha molts enviaments al mateix temps, el servidor triga més temps a executar les partides, i podria ser massa tard!
- No assumiu que el codi que vau enviar per última vegada és el mateix utilitzat pel nostre servidor per jugar les partides oficials. Assegureu-vos de triar el codi que voleu que us representi.
- Recordeu: manteniu el vostre codi senzill, compileu sovint, proveu sovint. O ho lamentareu.