Battle Royale II: Requiem

requiem.jutge.org

Omer Giménez Jordi Petit Salvador Roura

7 de novembre de 2011

1 Introducció

Hola a tothom! Aquest quadrimestre els alumnes d'EDA i d'ADA de la FIB i d'Algorísmia de la FME participaran en el joc **Battle Royale 2: Requiem**. Quina sort! Felicitats!

Les regles del joc són molt senzilles: els programes dels estudiants lluitaran els uns contra els altres durant diversos dies, a raó d'unes quantes rondes per dia. Al final de cada ronda s'eliminarà l'estudiant amb pitjors resultats. Quan més trigui un estudiant a ser eliminat, més alta serà la seva nota. Els últims 11 supervivents d'EDA i d'ADA, i els últims 5 supervivents d'Algorísmia obtindran la nota màxima i es classificaran per a la Gran Final, en la qual es disputaran diverses rondes eliminatòries fins a trobar el campió de **Battle Royale 2: Requiem**.

La Gran Final es celebrarà el dimarts 20 de desembre de 14.00 a 15.00h a la Sala d'Actes de la FIB. Tothom hi serà benvingut.

Al final, només en pot quedar un!

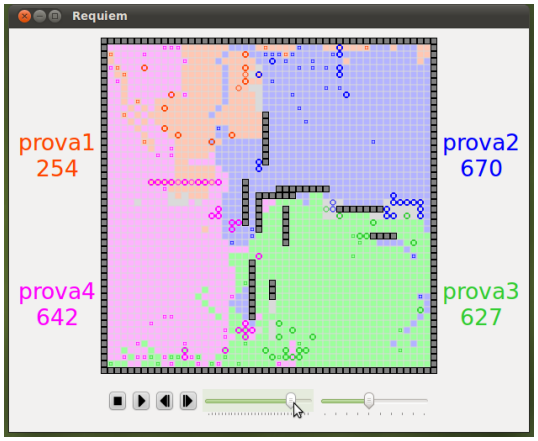
2 Regles del joc

- Quatre equips de robots, cadascun controlat per un programa, s'enfronten sobre un taulell quadrat amb 50×50 caselles. L'objectiu del joc és colonitzar el màxim nombre de caselles, tot pintant-les amb el color del propi equip.
- Cada equip té assignat un quadrant del taulell: els equips 1, 2, 3 i 4 tenen, respectivament, el quadrant NO (nord-oest), NE (nord-est), SE (sud-est) i SO (sud-oest).
- Hi ha dos tipus de robots: els *pagesos* i els *cavallers*. En començar la partida cada equip té 20 robots de cada tipus, situats aleatòriament a les caselles sense murs del quadrant de l'equip.
- Una partida dura 300 torns. A cada torn, els quatre programes ordenen els moviments dels seus robots. Cada robot pot intentar moure's a una de les caselles adjacents, ja sigui horitzontalment, verticalment, o en diagonal. Totes les ordres de tots els equips s'executen al final del torn, en ordre aleatori.
- Algunes caselles tenen murs, pels quals no es pot passar. En particular, tot el perímetre del taulell està envoltat de murs. Un robot no es mou en un torn si no rep cap ordre, o si en el moment d'intentar executar la seva ordre, la casella a què es dirigeix està ocupada, ja sigui per un altre robot o per un mur.
- Quan un robot pagès es mou, colonitza la casella que passa a ocupar, pintant-la amb el color del seu equip. Totes les caselles, exceptuant aquelles que tinguin inicialment un pagès a sobre, comencen sense cap color.
- Un robot cavaller ataca qualsevol robot rival que es trobi a la casella on s'intenta moure. Si el robot atacat és un pagès, aquest mor immediatament. Si el robot atacat és un cavaller, aquest perd aleatòriament entre 50 i 190 punts de vida. Els cavallers comencen amb 400 punts de vida, i moren quan quedarien amb 0 o menys punts. Mori o no el robot atacat, l'agressor no es mou en aquell torn. Un cavaller no ataca mai un robot del seu mateix equip.
- Un cavaller que no rebi cap ordre durant un torn (o que rebi l'ordre de no moure's) recupera 10 punts de vida al final del torn, després de tots els moviments i atacs, fins al màxim de 400 punts.
- Cada robot té un enter que l'identifica de manera única. Quan un robot mor, reneix formant part de l'equip que l'ha mort, totalment sa i amb un nou identificador que no s'hagi usat mai per cap robot. Així doncs, durant tota la partida hi ha sempre 40 robots de cada tipus.

- Al final de cada torn, els robots morts són retirats d'on estaven i reneixen a una casella lliure del quadrant del seu nou equip. Es tria automàticament una casella tan propera al centre del quadrant com sigui possible, i que no sigui contigua a cap robot rival. En el cas improvable que totes les caselles del quadrant fossin contígües a algun robot rival, es triaria simplement una casella lliure tan propera al centre del quadrant com sigui possible.

- Si un robot rep més d'una ordre durant un torn, només intenta executar la primera ordre donada.

- La puntuació d'un equip és el nombre de caselles colonitzades amb el seu color al final de l'últim torn. Un equip que acaba la partida sense caselles ni robots rep una puntuació de -1. Un equip que no acaba la partida perquè el programa que el controla no respon prou ràpid rep una puntuació de -2.



3 Programació

En el web del joc (`https://requiem.jutge.org`) trobareu el material necessari per programar el joc en C++. En concret, hi ha un simulador i un visor del joc, amb els quals podreu provar els vostres jugadors. Necessitareu Linux amb g++ i la llibreria QT versió 4.

Els ordinadors de la FIB i de la FME tenen aquest software instal·lat. En Ubuntu, amb una instrucció

```
sudo apt-get install build-essential libqt4-dev
```

tindreu tot el que us cal. En principi, el codi del simulador no fa servir res específic d'Ubuntu, de manera que també hauria de funcionar directament sobre altres sistemes operatius.

3.1 Com fer un jugador

Per fer un jugador, copieu el fitxer `AInull.cc` o `AIDemo.cc` a un fitxer `AIxxx.cc`, on `xxx` sigui un nom de la vostra elecció (mínimament respectuós, que no hagi estat triat ja per un altre estudiant, i compost per lletres, dígits i caràcter subratllat), per exemple `Terminator`. Fixeu-vos que, al fitxer `AITerminator.cc` (o com es digui) que acabeu de crear, heu de canviar la línia 7 per posar el nom del vostre jugador (a l'exemple, `#define NAMEPLAYER Terminator`, sense el prefix `AI`). També veureu que heu d'implementar el mètode `fes_torn()` de la classe `AIstruct`. Aquest mètode es crida una vegada per torn per conèixer les ordres que el vostre jugador (que de fet no és un programa complet, sinó un procediment) dóna als seus robots.

Per provar el vostre jugador heu de fer el següent, en aquest ordre:

- `qmake`
  Detecta si hi ha nous fitxers `AI*.cc` en el directori actual, i crea un fitxer `Makefile` per compilar-los. Executeu `qmake` només quan afegiu un nou jugador `AI*.cc`. No cal executar-lo quan modifiqueu algun `AI*.cc`.

- `make`
  Compila el simulador, el visor de partides i tots els jugadors `AI*.cc` que hi hagi en el directori actual, i crea un únic executable `Requiem` que ho conté tot.

- `./Requiem null Terminator null Terminator 123`
  Aquest exemple executa una partida, on els equips 1, 2, 3 i 4 estan controlats pels jugadors `null`, `Terminator`, `null` i `Terminator`, respectivament, i on la llavor aleatòria inicial és 123. Escrivint `./Requiem` podreu veure tots els jugadors que conté, i les opcions d'ús.

Els jugadors que envieu al web del joc han de complir certes limitacions: Han d'estar en un sol fitxer. No poden fer servir variables globals (si voleu guardar informació d'un torn per al següent, haureu d'afegir atributs estàtics a la classe `AIstruct`). En un ordinador "normal", no poden trigar més d'una dècima de segon per torn, ni més de 10 segons en el total d'una partida. No poden fer servir el `cin` ni el `cout`, però sí que poden fer servir el `cerr`. Només poden usar les llibreries estàndard de C++, com ara `vector`, `map`, etc. No poden obrir fitxers ni fer altres crides al sistema operatiu (threads, forks, etcètera).

3.2 Com funciona el taulell de joc

El taulell del joc és una matriu 50×50, on cada casella s'indexa amb dos nombres `(i,j)` entre 0 i 49 que n'indiquen la fila `i` i la columna `j`. Nord i Sud volen dir, respectivament, decrementar i incrementar la fila `i`. Est i Oest volen dir, respectivament, incrementar i decrementar la columna `j`.

Quan programeu, suposeu que sou l'equip 1 (el del quadrant Nord-Oest), perquè el simulador rota internament els taulells i fa els canvis necessaris per

fer creure a cada jugador que controla l'equip 1. En particular, podeu suposar que els vostres robots començaran la partida dins de les posicions $[1, 23] \times [1, 23]$, amb els enemics situats a l'Est, al Sud-Est, i al Sud.

Com ja s'ha dit, s'ha de programar un mètode `void fes_torn()` que serà cridat una vegada a cada torn. Per conèixer la situació del taulell i dels robots a cada moment, i per donar ordres als vostres robots, haureu d'usar els següents mètodes:

- `int torn_actual()`: Retorna el torn actual. Val 1 si és el primer torn. Val 300 si és l'últim torn.
- `int color(int i, int j)`: Retorna el color entre 1 i 4 amb què està pintada la casella `(i,j)`, 0 si encara no ha estat pintada, o -1 si la casella té un mur.
- `bool existeix(int i, int j)`: Retorna si hi ha un robot a la casella `(i,j)`.
- `t_robot robot(int i, int j)`: Retorna la informació del robot situat a la casella `(i,j)`. Si no n'hi ha cap, retorna un robot "nul".

  Un `t_robot` es un struct amb diversos camps de tipus enter: `id` (identificador), `i`, `j` (la posició actual), `model` (1 per als pagesos, 2 per als cavallers), `equip` (entre 1 i 4), i `vida` (els punts de vida, que sempre són 0 per als pagesos, i estan entre 1 i 400 per als cavallers). Un robot "nul" té tots els camps a -1.
- `vector<int> pagesos(int e)`: Retorna un vector amb els identificadors dels robots pagesos de l'equip `e`.
- `vector<int> cavallers(int e)`: Retorna un vector amb els identificadors dels robots cavallers de l'equip `e`.
- `bool existeix(int id)`: Retorna si existeix un robot amb identificador `id`.
- `t_robot robot(int id)`: Retorna la informació del robot amb identificador `id`. Retorna un robot "nul" si ara no existeix cap robot amb aquest identificador.
- `int num_caselles(int e)`: Retorna el nombre de caselles pintades amb el color `e`. Podeu usar -1 per a les caselles amb murs, i 0 per a les caselles encara sense pintar.
- `int random(int n)`: Retorna un nombre pseudo-aleatori entre 0 i `n` $- 1$.
- `bool ordena(int id, int dir)`: Ordena al robot amb identificador `id` que es mogui (o ataqui) en la direcció `dir`. Retorna *cert* si l'ordre donada és vàlida, és a dir, si hi ha un robot del propi equip amb aquest identificador, si és la primera ordre que se li dóna en aquest torn i si el robot,

seguint la direcció indicada, no xocaria contra un mur ni sortiria fora de limits. Hi ha nou direccions possibles, entre 1 i 9, que es corresponen respectivament a `EST`, `SE`, `SUD`, `SO`, `OEST`, `NO`, `NORD`, `NE`, i `RES` (estar-se quiet). Aquestes constants ja estan definides.

Sempre que es crida un mètode amb paràmetres incorrectes (una casella fora del taulell, una `e` que no està entre 1 i 4 a `pagesos(e)` o a `cavallers(e)`, una `e` que no està entre -1 i 4 a `num_caselles(e)`, una `n` no estrictament positiva a `random(n)`, o quan `ordena(id, dir)` retorna *fals*) la classe `Mapa` retorna -1 si el tipus a retornar era un enter, un robot "nul" si el tipus a retornar era un `t_robot`, i un vector buit si el tipus a retornar era un vector. A més, s'escriu un missatge d'error per la consola. Cridar mètodes amb paràmetres incorrectes no provocarà mai que el vostre programa es pengi, però tingueu en compte que el pot fer anar bastant més lent, degut a l'escriptura dels missatges.

3.3 Com gravar i veure partides

Per guardar el resultat d'una partida, afegiu el paràmetre `--out=fitxer.req` quan executeu `Requiem`, això crearà el fitxer `fitxer.req` amb el contingut de la partida. El web del joc també us donarà fitxers `.req` per veure les partides disputades al servidor.

Per visualitzar una partida ja guardada, feu `Requiem fitxer.req`.

4 El web del joc

Al web de Requiem és a `https://requiem.jutge.org` i és on podreu trobar aquestes instruccions i el codi necessari per començar a jugar. Haureu de fer servir aquest web per fer els lliuraments. També el podreu usar per jugar partides on-line contra els programes d'altres estudiants. I tothom podrà veure els estudiants eliminats en temps real.

És normal que el vostre navegador es queixi al accedir al web de Requiem en mode de connexió segura perquè el certificat no està comprovat per un tercer.

4.1 Lliuraments

Podeu fer més d'un programa amb estratègies diferents. Sempre que es vulgui es podrà pujar al web versions actualitzades dels vostres fitxers `AI*.cc`. De forma automàtica, es farà el procés de verificació següent:

- Es jugarà una partida amb quatre còpies del programa enviat. Si algun dels programes triga massa temps a respondre, l'enviament es rebutjarà.
- Es jugarà quatre partides contra tres programes tontos (un programa tonto és un jugador programat pels organitzadors amb una estratègia molt

senzilla). El programa enviat serà rebutjat si no guanya les quatre partides.

En el cas que un programa sigui rebutjat, se n'indicarà el motiu, i es mostrarà la raó o la partida causant del rebuig. En cas contrari, el programa s'acceptarà al web del joc, podrà disputar partides on-line, i el seu nom estarà reservat: ningú podrà tornar a enviar un altre programa amb el mateix nom. En qualsevol cas, els vostres codis no seran mai mostrats als altres estudiants.

4.2 Partides d'entrenament

El web del joc permet celebrar partides on-line entre els programes acceptats. Aquestes partides es consideren "d'entrenament" perquè els seus resultats, que només són visibles als autors dels programes, no afecten la classificació.

Per fer-ho, es té el concepte de participants *amics*, i es distingeix entre programs d'ús *privat* i d'ús *públic*. Per defecte, els programes enviats són d'ús privat: només l'autor i els seus amics poden jugar partides on-line amb aquells programes. En concret, un programa privat no pot participar en una partida si no és que el seu autor és amic de tots els autors dels programes de la partida.

Es pot evitar aquesta restricció fent que un jugador sigui d'ús públic. Fent això, es podrà enfrontar contra tots els altres jugadors d'ús públic. Quan un jugador es fa d'ús públic, no es permet tornar-lo a fer d'ús privat.

La única excepció és el programa Tonto (de l'autor `salvador.roura`) que s'ha d'haver guanyat per a ser acceptat: qualsevol programa acceptat, sigui públic o privat, pot enfrentar-se contra el Tonto.

Mitjançant el web es pot modificar l'ús dels jugadors, demanar que es juguin partides on-line, i administrar la llista dels amics. Perquè dos participants es considerin amics és necessari que ambdós s'incloguin mútuament a les seves llistes.

4.3 Partides oficials

El joc començarà el dilluns 7 de novembre cap les 11:00. El web estarà en funcionament des del primer moment. Per participar a la pràctica, caldrà tenir un programa acceptat abans del divendres 25 de novembre a les 15:00. Els participants podran triar quin dels seus programes acceptats el representarà a les partides oficials.

A partir del dilluns 26 de novembre, si no hi ha imprevistos, el web anirà organitzant rondes amb els programes representants dels estudiants supervivents. Els resultats d'aquestes partides i les pròpies partides seran visibles per a tothom.

A cada ronda, s'eliminarà a l'autor del programa amb pitjors resultats, seguint el procés següent: s'agrupen els programes de quatre en quatre a l'atzar, afegint programes tontos de reserva si convé per arribar a un múltiple de 4. Per a cada grup de 4, es juga una partida. En cada partida, se salva qualsevol programa que obtingui millor puntuació que dos dels seus tres rivals a

la partida. Es repeteix el procés (agrupar de 4 en 4, afegir programes tontos, etc.) amb els programes no salvats, fins que únicament quedin 4 o menys programes. Aleshores, es disputen partides entre aquests programes (afegint, si convé, programes de reserva) fins que un d'ells quedi per darrere dels seus rivals, sense comptar els de reserva. Aleshores, l'autor del programa perdedor queda eliminat.

Cada estudiant podrà triar quin dels seus programes acceptats l'ha de representar, i podrà canviar aquesta decisió tantes vegades com vulgui, en particular si envia versions millorades dels seus programes. Mentre es disputi una ronda, no es podrà canviar el programa: el canvi es farà efectiu en començar la propera ronda (si no s'ha quedat eliminat).

4.4 Novetats i incidències

Qualsevol novetat o qualsevol incidència que els organitzadors del joc vulguin donar a conèixer es publicarà en el web del joc.

4.5 Resultats

El del joc mostrarà a tothom i en temps real els resultats de cada ronda.

4.6 Fòrum

El fòrum del joc pels estudiants matriculats d'EDA es troba al Racó.

5 Normativa

El mètode d'avaluació del joc es troba publicat a la Guia Docent de les assignatures pertinents.

El lliurament de programes pel joc implica acceptar que, si es comprova que hi ha hagut frau en la seva realització, es restaran els seus punts (enlloc de sumar-los) als de l'assignatura.

Els programes enviats es compararan periodicament entre ells i els enviats en cursos anteriors per detectar possibles còpies.

Davant de qualsevol imprevist, els professors prendran les mesures que considerin més oportunes.

6 Consells

Us recomanem seguir els consells següents:

- Comenceu amb estratègies molt senzilles, que siguin fàcils de programar i de depurar, que és exactament allò que necessiteu al principi.

- Programeu procediments senzills i útils, i *assegureu-vos que funcionin correctament*.
- No us arrisqueu a ser eliminats abans de començar. Aconseguiu que el web us accepti un jugador tan ràpidament com us sigui possible. Un jugador acceptat representa tenir assegurada part de la nota de la pràctica!
- Conserveu diverses versions antigues (però que funcionin correctament) dels vostres jugadors, i no les toqueu per res.
- Compileu sovint, testejeu sovint. És *molt* més fàcil trobar i corregir els errors quan s'han canviat unes poques línies de codi que quan s'acumulen molts canvis de cop.
- Feu servir `cerr`s per posar xivatos, i `assert`s per comprovar que el codi fa el que toca. Però comenteu els `cerr`s abans d'enviar el jugador al web, ja que alenteixen els programes.
- Per testejar un jugador nou, enfronteu-lo contra uns altres tres jugadors que no treguin cap missatge. Així només apareixeran per pantalla els missatges del nou jugador.
- Un error de segmentació en el vostre codi aturarà el simulador, i no podreu veure què ha provocat l'error del jugador. Apreneu a usar el `valgrind` si no sou capaços de corregir els errors amb `cerr`s.
- No us arrisqueu a quedar eliminats per culpa d'un jugador massa lent. Doneu-vos un bon marge de seguretat. En particular, eviteu tant com pugueu l'escriptura de missatges.
- Feu competir els vostres jugadors contra rivals diferents, i estudieu les partides. Encara que no podreu veure els codis dels altres estudiants, si sou capaços de deduir les seves tàctiques i us semblen útils, podeu mirar d'imitar-les, defensar-vos-en o millorar-les.
- No deixeu el vostre codi a ningú, ni tan sols d'una versió antiga. Un cop feta la competició s'analitzaran els programes de cada ronda amb programes detectors de plagi (JPlag, per exemple). També es comparan els jugadors d'aquest curs amb els jugadors de cursos anteriors.
- Per evitar disgustos, feu servir el web de Requiem per jugar partides amb els vostres amics. Però no confieu massa en els vostres amics...
- Tingueu en compte que podeu lliurar nous jugadors en qualsevol moment (excepte durant la fase final).
- No espereu fins al darrer moment per enviar les vostres solucions. Els demés també ho fan i la cua del Jutge no dóna abast.
- Insistim: Feu codis senzills. Compileu sovint, testejeu sovint. O afronteu-ne les conseqüències.