# Proiectarea Algoritmilor - Tema 2

Alexandru Damian

April 1, 2012

- **14 aprilie 2012:** Au fost adaugate explicatii suplimentare in sectiunea *Protocol*
- **12 aprilie 2012:**
  - A fost clarificat un exemplu din sectiunea *Scoaterea pieselor*
  - Au fost relaxati timpii minimi si maximi pentru punctare (5, respectiv 15 secunde).
  - Binarele au fost modificate:
    - au fost adaugati cei trei boti
    - s-a adaugat inca un parametru la server, pentru marirea timeout-ului. Exemple de rulare:
      - *java -jar server.jar*: port default 6666, timeout default 30 de secunde
      - *java -jar server.jar 7777*: port 7777, timeout default 30 de secunde
      - *java -jar server.jar 7777 100*: port 7777, timeout 100 de secunde
      - **comanda *java -jar server.jar 100* va avea efectul port 100, timeout default 30 de secunde si nu va functiona**
    - serverul afiseaza numarul de duble, zaruri de diferenta 2 si suma totala a zarurilor pentru un joc, precum si punctajul obtinut
- **4 aprilie 2012:**
  - S-a corectat o greseala in enunt, referitoare la pozitia caselor jucatorilor (*sectiunea **Tabla de joc***), precum si modul de readucere a pieselor de pe bara inapoi pe tabla (*sectiunea **Mutarea unei piese**, ultimul paragraf*).
  - La *sectiunea **Scoaterea pieselor***, am punctat un exemplu mai relevant.
  - La *sectiunea **Protocol***, am redenumit campul **tox** dintr-un mesaj in **spanx**, pentru o claritate sporita.
- **3 aprilie 2012:** Protocolul a fost modificat, prin adaugarea unui nou mesaj. Schimbarile sunt colorate in rosu.

# 1 Introducere

Dupa ce au terminat cu succes prima tema, studentii de la Proiectarea Algoritmilor au decis ca e timpul sa ia o pauza de joaca. Ei vor acum sa devina experti in jocul numit **Backgammon** (sau, mai cunoscut, **Table**). Norocul le surade, pentru ca aceasta tema va avea drept cerinta implementarea unui program care sa *stie* sa joace acest joc.

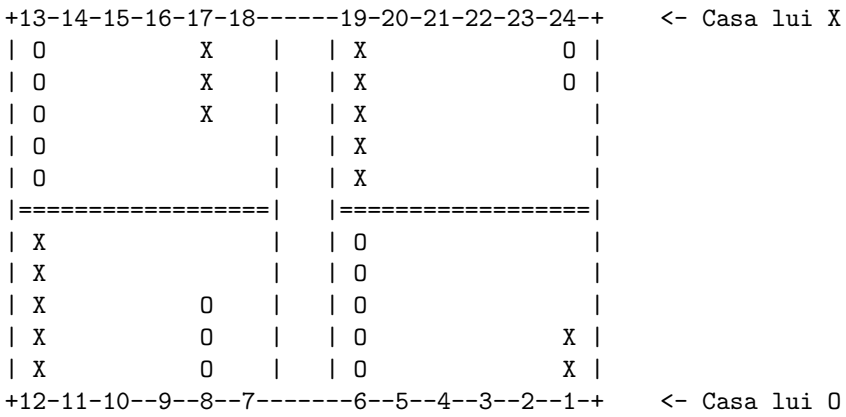
# 2 Regulile jocului

## 2.1 Tabla de joc

Tabla de joc este impartita in 5 zone, astfel:

- 2 zone asezate vis-a-vis, ce reprezinta casele jucatorilor;
- 2 zone asezate vis-a-vis, ce reprezinta zone de tranzit;
- cele 2 perechi de zone sunt despartite de o fasie verticala numita bara.

Exista deci mai multe variante echivalente pentru tabla de joc. Noi o vom folosi pe urmatoarea:




Pe aceasta tabla, zona numerotata **1 - 6** va fi casa jucatorului **O (alb)**, iar zona numerotata **19 - 24** va fi casa jucatorului **X (negru)**. Mutarile pieselor unui jucator se vor realiza dinspre casa adversarului, spre casa proprie. Astfel, jucatorul **X** va muta in sens crescator, in sens orar (de la 1 spre 24), iar jucatorul **O** va muta in sens descrescator, in sens trigonometric (de la 24 spre 1).

## 2.2 Scopul jocului

Scopul jocului este de a duce toate piesele proprii in casa, urmand ca apoi sa fie scoase. Primul jucator care scoate toate piesele proprii de pe tabla castiga.

## 2.3 Determinarea mutarilor disponibile

Mutarile disponibile, precum si numarul lor sunt determinate de aruncarea a doua zaruri, astfel:

1. In cazul obtinerii a doua valori diferite, sunt disponibile doua mutari, pe o distanta egala cu valoarea inscrisa pe zar. De exemplu, pentru aruncarea de zaruri **(3, 5)**, jucatorul va avea la dispozitie doua mutari, una de **3 spatii**, iar cealalta de **5 spatii**.

2. In cazul obtinerii unei duble, sunt disponibile patru mutari, fiecare pe o distanta egala cu valoarea inscrisa pe oricare dintre zaruri. De exemplu, pentru aruncarea de zaruri **(3, 3)**, jucatorul va avea la dispozitie patru mutari, fiecare de **3 spatii**.

## 2.4 Mutarea unei piese

**Nota!** Se considera ca, atunci cand se realizeaza o mutare de **k** spatii, piesa respectiva va **sari peste toate cele k-1 spatii** pentru a ajunge la destinatie (i.e. conteaza doar starea spatiilor sursa si destinatie).

O piesa se poate muta intr-un spatiu destinatie doar daca starea acestuia este una din urmatoarele (ilustram pentru jucatorul **alb**).

- spatiul este gol. In acest caz, acest spatiu va fi ocupat de jucatorul alb, cu o piesa.
- spatiul este ocupat de jucatorul alb. In acest caz, spatiul va ramane ocupat de jucatorul alb, care isi va adauga inca o piesa pe acesta.
- spatiul este ocupat de jucatorul negru, dar **cu o singura piesa**. In acest caz, spatiul este cucerit de jucatorul alb. Acesta va scoate piesa neagra pe bara si va ocupa spatiul cu propria piesa.

Pentru a readuce o piesa in joc (a o scoate de pe bara), jucatorul respectiv va trebui sa execute o mutare valida de pe *spatiul imaginar* de la granita exterioara a casei adversarului (i.e. coloana 0 pentru jucatorul negru, sau coloana 25 pentru jucatorul alb)

## 2.5 Scoaterea pieselor

Atunci cand toate piesele voastre se afla in casa, puteti incepe sa le scoateti. O piesa se poate scoate de pe un spatiu alfat la k pozitii de marginea exterioara a casei doar cu o mutare de k spatii. In cazul in care exista doar piese mai apropiate de marginea exterioara decat k spatii, se vor scoate piesele in ordinea distantei.
Exemple, pentru jucatorul **alb**, a carui casa se afla in pozitiile **1 - 6**:

- Daca zarurile aruncate sunt **(3, 5)**, iar tabla mai are cate o piesa pe fecare spatiu din casa, se pot face urmatoarele mutari:
  - mutarea piesei de pe spatiul 6 pe spatiul 3 (cu zarul 3) si scoaterea piesei de pe spatiul 5
  - scoaterea pieselor de pe spatiile 3 si 5 s.a.
- Daca zarurile aruncate sunt **(3, 5)**, iar tabla mai are doar doua piese pe pozitia 4, se poate face doar urmatoarea mutare: cu zarul 5 se scoate piesa de pe spatiul 4, iar cu zarul 3 se muta piesa de pe spatiul 4, pe spatiul 3.

## 2.6 Restrictii

- **Nu** se pot executa mutari cu piesele de pe tabla daca exista piese scoase pe bara. Acestea trebuie reintroduse primele pe tabla.
- **Nu** exista o limita asupra numarului de piese care pot fi scoase pe bara.
- Este posibil ca unele mutari disponibile sa fie imposibil de executat (i.e. aveti 1 piesa pe bara, casa adversarului are doar un spatiu neaparat de 2 sau mai multe piese, iar zarurile voastre nu au acea valoare). In acest caz, mutarea **va fi pierduta**. Totusi, **este obligatoriu sa executati toate mutarile posibile, chiar daca va dezavantajeaza**.

# 3 Cerinta

Se cere implementarea unui program care sa fie capabil sa joace corect si sa castige jocul Backgammon.

## 3.1 Implementare

Cele doua programe adverse se vor conecta la un server care va superviza jocul. Acest server va decide cine va juca cu piesele albe si cineva va juca cu piesele negre, va mentine starea tablei si ii va determina consistenta, iar in final va anunta castigatorul si pierzatorul.

## 3.2 Desfasurarea jocului

Un program (client) se va conecta la server printr-un socket, iar comunicatia se va face sincron, astfel:

1. Clientul trimite serverului un mesaj prin care il anunta care ar trebui sa fie dificultatea oponentului sau.
2. Clientul primeste un mesaj prin care afla daca el este jucatorul negru sau jucatorul alb.
3. Clientul primeste un mesaj prin care afla mutarea adversarului si zarurile "aruncate" de server pentru mutarea viitoare. (In cazul in care e prima mutare a clientului alb, atunci mutarea adversarului va fi vida)
   **Toate mutarile primite de la oponent prin intermediul serverului se presupun a fi corecte.**
   La acest pas, clientul poate primi un alt tip de mesaj, prin care e notificat daca a castigat sau a pierdut.
4. Clientul executa mutarile primite folosind piesele adversarului si realizeaza noile mutari asociate zarurilor lui.
5. Clientul trimite mutarile noi catre server, care le verifica si le trimite mai departe oponentului.
   - In cazul in care o mutare este incorecta sau nu s-au executat toate mutarile posibile, jucatorul care a gresit pierde.
   - In cazul in care jucatorul a scos toate piesele de pe tabla, el castiga.
   - In cazul in care mutarea jucatorului dureza peste 30 de secunde, el pierde.

## 3.3 Protocol

Vom detalia in continuare modul de codificare a mesajelor.
Toate mesajele care se vor transmite vor fi **siruri de octeti** de dimensiune fixa. Mesajele se vor transmite astfel: **[D][mesaj]**, unde D este **un octet** ce

reprezinta **dimensiunea mesajului**.

Mesaje primite de client (pasii (2) si (3) de mai sus):

- *Mesajul prin care un client afla daca e negru sau alb:*
  Este un mesaj de lungime **1 octet**, ce va contine valoarea **0, daca clientul este alb si 1, daca clientul este negru**. Un exemplu de astfel de mesaj trimis de catre server este: **(1, 1)**

- *Mesajul prin care clientul afla mutarile executate de adversar, precum si valoarea zarului:*
  Este un mesaj de lungime **para**, astfel:
  **[from1][span1][from2][span2]...[fromk][spank][dice1][dice2].** Astfel, pentru **k** mutari, lungimea mesajului va fi **2k + 2**. Un exemplu de astlfe de mesaj trimis de catre server este: **(6, 3, 4, 2, 1, 5, 5)**. Semnificatia lui este urmatoarea: lungimea mesajului este 6, deci exista 2 mutari executate de adversar: (3, 4), respectiv (2, 1). Valorile zarurilor sunt (5, 5). O mutare este codificata astfel: **(de unde plec, cat merg)**. Mai exact, octetii **fromx** au valori intre 1 si 24, precum si valoarea 30 (BAR). Octetii **spanx** au valori intre 1 si 6.

- *Mesajul primit de un client atunci cand castiga sau pierde (in locul mesajului cu mutarile adversarului si zarurile proprii):*
  Este un mesaj de lungime **1 octet**, ce va contine valoarea **87 ('W'), daca clientul este castigator si 76 ('L'), daca clientul este infrant**. Un exemplu de astfel de mesaj trimis de catre server este: **(1, 87)**

Mesaje trimise de client (pasii (1) si (5) de mai sus):

- *Mesajul prin care clientul anunta serverul care ar trebui sa fie dificultatea oponentului sau:*
  Este un mesaj de lungime **1 octet**, care poate lua urmatoarele valori:

  - **1**, pentru un client care nu are nicio tactica si face mutari doar pentru a se apropia de casa, fara sa atace sau sa se apere;
  - **5**, pentru clientul de maxim 5 puncte din 8;
  - **7**, pentru clientul de maxim 7 puncte din 8;
  - **8**, pentru clientul de maxim 8 puncte din 8;

  De exemplu, pentru a juca impotriva adversarului de nota 7, veti trimite urmatorul mesaj catre server: **(1, 7)**

- *Mesajul prin care clientul anunta serverul despre mutarile pe care le poate face:*
  Este un mesaj de lungime **para**, astfel:
  **[from1][span1][from2][span2]...[fromk][spank].** Astfel, pentru **k** mutari, lungimea mesajului va fi **2k**. Un exemplu de astlfe de mesaj trimis catre server este: **(6, 2, 2, 2, 2, 5, 6)**. Semnificatia lui este urmatoarea:

lungimea mesajului este 6, deci exista 3 mutari ce se doresc a fi executate: (2, 2), (2, 2), respectiv (5, 6).
*De notat ca aceasta serie de mutari este incorecta si va duce la pierderea jocului, pentru ca, pentru a fi permise trei mutari, este necesar ca zarul sa fie o dubla, a patra mutare sa fie imposibil de realizat si toate cele trei mutari sa fie pe aceeasi distanta (in cazul nostru, 2 != 6)*
Octetii **fromx** au valori intre 1 si 24, precum si valoarea 30 (BAR). Octetii **spanx** au valori intre 1 si 6.
Pentru clarificare, veti trimite pe socket catre server sirul de octeti **[D][mesaj]** sau, folosind formatul de mai sus, **[2k][from1][span1]...[fromk][spank]**. Deci, pentru mutarile (2, 2), (2, 2), (5, 6), mesajul util este [2, 2, 2, 2, 5, 6], iar mesajul trimis catre server este [6, 2, 2, 2, 2, 5, 6].

# 4 Punctare

Vi se vor pune la dispozitie trei clienti cu niveluri diferite de dificultate:

- un client de dificultate **scazuta**, pentru infrangerea caruia veti putea obtine **maxim 5 puncte**;
- un client de dificultate **medie**, pentru infrangerea caruia veti putea obtine **maxim 7 puncte**;
- un client de dificultate **crescuta**, pentru infrangerea caruia veti putea obtine **maxim 8 puncte**;

In plus, vi se va oferi **maxim 1 punct** pentru **README si explicatii** si **maxim 1 punct** pentru **coding style**.
Fractiunea din 8 puncte pe care o veti primi se va calcula in felul urmator:

1. se va alege clientul cel mai dificil pe care il puteti bate in 2 meciuri din 3, pentru care se pot obtine **MAX** puncte
2. se va calcula timpul mediu petrecut pe mutare (in secunde); se va rotunji inferior la 5 si superior la 5, obtinand valoarea **T**
3. se va calcula punctajul final (din maxim 8) cu formula $(1.5 - 0.1T) * MAX$

De exemplu, presupunem ca bateti doar 1 meci din 3 impotriva clientului de dificultate crescuta si bateti toate meciurile impotriva clientului de dificultate medie. In acest caz, $MAX = 7$. Presupunem ca timpul mediu per mutare este de 10 secunde, deci $T = 5$. Punctajul final obtinut este: $(1.5 - 0.1 \cdot 10) \cdot 7 = 3.5 puncte$

**Va rugam sa notati in README cel mai dificil client pe care il puteti invinge.**