KOOPERÁCIÓ ÉS GÉPI TANULÁS LABORATÓRIUM

Szavazás és Aukció

Elméleti segédlet

Készítette:
Kovács Dániel László
(dkovacs@mit.bme.hu)

Méréstechnika és Információs Rendszerek Tanszék
Budapesti Műszaki és Gazdaságtudományi Egyetem

2012, október.

Bevezető

Ebben az anyagban a *Méréstechnika és Információs Rendszerek Tanszék* által indított „*Intelligens Rendszerek*” M.Sc. szakirány „*Kooperáció és Gépi Tanulás (VIMIM223)*” szakiránylaboratóriumának „***Szavazás és Aukció***” c. laborgyakorlata szoftveres infrastruktúráját dokumentáljuk. Lényegében 4 szoftverágens bemutatása fog megtörténni:

- **SZAVAZÁS**
  - `VoterAgent`: **szavazó-ágens** többségi szavazáshoz
  - `VotingMechanismAgent`: **szavazásvezérlő-ágens** többségi szavazáshoz
- **AUKCIÓ**
  - `BidderAgent`: **licitáló-ágens** angol aukcióhoz
  - `AuctioneerAgent`: **árverező ágens** angol aukcióhoz

Szavazás

A hallgatók feladata a gyakorlat szavazás része során olyan szavazó (továbbiakban: ***Voter***) ágensek létrehozása, melyek képesek különböző szavazásokban részt venni. A szavazásokat egy szavazásvezérlő (továbbiakban: ***VotingMechanism***) ágens vezeti.

A hallgatók rendelkezésére kezdetben egy egykörös többségi szavazásra alkalmas *Voter* és *VotingMechanism* ágenst bocsájtunk. Ezek alkalmas átírásával kell a különböző szavazásokat (pl. többségi 2-körös runoff szavazás (plurality runoff), többségi elven (majority rule) működő szavazás, engedélyező szavazás (approval), pontozó szavazás (rated), Borda-féle rangsorolás (ranked Borda count)) megvalósítaniuk. A *Voter* ágenseket összevetjük a szavazás végén, hogy megállapítsuk, hogy pl. több körös esetben melyiknek a legjobb a szavazási stratégiája.

Részletes specifikáció

Két fajta ágens fut majd tehát egy JADE platformon: **(1)** *VotingMechanism*, és **(2)** *Voter*. Mindkettő 1-körös sima többségi szavazásra (simple plurality voting) készen. A hallgatók feladata ezek átírása.

Egy *VotingMechanism* és több *Voter* ágens tehát egy sima többségi szavazásban vesz részt, amihez a *VotingMechanism* ágens számára a priori adott egy (lokálisan hozzáférhető/letölthető) leírás. A leírás tartalmazza, hogy milyen opciókra lehet szavazni. Ezeket az adatokat tehát a *VotingMechanism* ágens induláskor megkapja egy megfelelő szövegfájl (TXT) formájában. A fejlesztéskor ennek megfelelően az alábbi formátumú szövegfájl megfelelő beolvasására kell felkészülni – ilyen egy **szavazás konfigurációja**:

```
O1 O2 … On
```

Itt tehát *n* darab adat szerepel szóközzel elválasztva:

- az *i*. opció [`Oi`], amire szavazni lehet, egy karakterfüzér.

Tipikusan legalább 2 opció lesz. Egy-egy *Voter*-nek egy körben csak egyetlen szavazási lehetősége van.

A *Voter* ágensek célja, hogy az általuk (vagy a felhasználójuk által) óhajtott kimenetel álljon elő a szavazás végeredményeképp. Ezt fejezi ki az alábbi képlet, pontosabban **jósági mérce**.

$$\text{Utility}(\text{Voter}) = \begin{cases} 0, & \text{ha az utolsó körben nem nyertesre szavazott} \\ 1, & \text{egyébként} \end{cases}$$

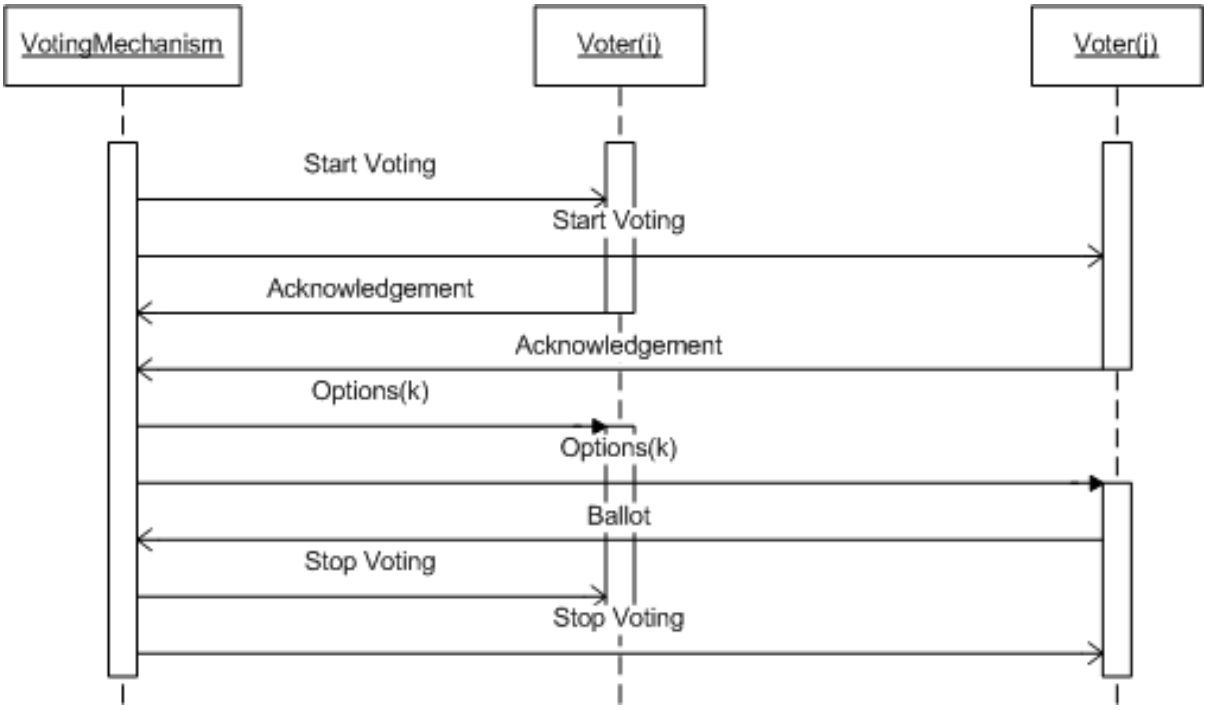
Az ágensek célja tehát ennek az *objektív*[1] haszonfüggvénynek a maximalizálása.

[1] Nyilván ettől még az ágensnek lehet egy saját *szubjektív* haszonfüggvénye, vagy rangsora/preferenciája is, ami meghatározza, hogy mikor, melyik körben melyik opcióra teszi le a voksát. De az ágensek sikerességét végső soron *külső szemlélőként*, a szavazók egyéni preferenciáinak ismerete nélkül másként megítélni nemigen lehet.

A szavazás menete (részletesen)

Mindként fajtájú ágens indulás után azonnal beregisztrál a DF ágensnél. A `ServiceDescription`-ben a *VotingMechanism* ágens „`votingmechanism`”-ot, a *Voter* ágens pedig „`voter`”-t ad meg típusként. A szolgáltatás neve is ugyanez mindkettőnél.

A platformon először a *Voter* ágensek indulnak, majd ezt követően az *VotingMechanism* ágens, amely indulás után lekérdezi a DF-től a platformon aktuálisan jelen lévő, regisztrált *Voter* ágensek listáját. Ezek után lényegében a következő üzenetváltás zajlik le majd le.



Itt lényegében arról van szó, hogy:

1. A ***VotingMechanism*** ágens kihirdeti, hogy elindult a szavazás.
    - Üzenet típusa: *INFORM*
    - Üzenet tartalma: *start*
        i. ahol a „*start*” egy sima karakterfüzér (és azt jelzi, hogy elindult a szavazás)
2. Erre a ***Voter***-ek egy beleegyezéssel válaszolnak.
    - Üzenet típusa: *AGREE*
    - Üzenet tartalma:
3. Erre a ***VotingMechanism*** ágens elkezdi körönként meghirdetni az adott körben még aktuális opciókat/alternatívákat, és várni a szavazatokat. A k-adik körben egy olyan broadcast üzenetet küld a *Voter*-ek számára, amiben jelzi, hogy a k-adik körről van szó, és hogy milyen opciókra lehet szavazni (kezdetben (k=1 esetén) ez nyilván egybeesik a fájlként adott szavazási konfigurációval). Ezek után adott idő elteltével (pl. 5 mp után), ha senki sem szavaz, akkor a (k+1)-edik kör következik (amíg el nem érjük a körök maximális számát, ugyanis akkor a szavazás leáll). Ha viszont valamelyik *Voter* (amelyik az adott körben még nem szavazatott) időközben szavaz (lásd. alább a REQUEST üzenetet), akkor az idő-számláló újraindul, és a beérkezett szavazatot a *VotingMechanism* ágens figyelembe veszi, azaz hozzáadja a az adott

opcióra eddig érkezett szavazatok számához (ami a kör kezdetén nyilván zérus). Ha már nem jön újabb szavazat a megadott időlimiten belül, vagy pedig már mindenki szavazott, akkor két lehetőség van: ha (1) egy olyan opció van, amelyre maximális számú szavazat érkezett, akkor az az opció nyer, ha pedig (2) nincs egyértelmű maximum, úgy az adott kör eldöntetlen (egy-körös szavazás esetén ez a szavazás eredményét is eldöntetlenné teszi). Az utolsó kör befejeztével a szavazás lezárul, és a *VotingMechanism* ágens kiírja a *Voter* ágensek ranglistáját.

- Üzenet típusa: *INFORM*
- Üzenet tartalma: *k* $option_1$ $option_2$ *...* $option_N$
    i. ahol a „*k*” egy zérónál nagyobb egész szám (és az aktuális kör számát jelzi – kezdetben 1)
    ii. „$option_i$” egy karakterfüzér (és az *i*. opciót jelzi)

4. Egy ***Voter*** ágens, amennyiben egy adott körben opciókról értesül, szavazhat (pl. ha a kezdetben bemenetként kapott alternatíva szerepel az aktuálisan felkínált opciók közt). Erre szolgál a következő, *VotingMechanism* ágensnek címzett üzenet.
    - Üzenet típusa: *REQUEST*
    - Üzenet tartalma: *0 0 0 ... 0 1 0 ... 0*
        i. ez a szóközökkel elválasztott, csupán 0-ákból és 1-esekből álló karakterfüzér mindössze egyetlen helyen tartalmaz 1-est (sima többségi szavazás esetén) – ott, ahol a *Voter* ágens számára indítási paraméterként átadott opció szerepel a *VotingMechanism* ágens által adott körben felkínált opciók között. Minden más helyen zérus van.
5. Amennyiben az utolsó szavazási kör végére értünk, vagy pedig már előbb eldőlt (egyértelművé vált) a szavazás eredménye, úgy a ***VotingMechanism*** ágens ezzel az üzenettel jelzi, hogy lezárja a szavazást.
    - Üzenet típusa: *INFORM*
    - Üzenet tartalma: *stop*
        ii. ahol a „*stop*” egy sima karakterfüzér (és azt jelzi, hogy lezárult a szavazás).

Miután lezajlott a szavazás, a *VotingMechanism* ágens jelzi (a tárolt és kiszámított adatok alapján), hogy a fenti jósági/hasznossági/sikerességi mércének megfelelően mi az egyes *Voter* ágensek konkrét haszna.

Példa

Tegyük fel a példa kedvéért, hogy 4 *Voter* ágens verseng: `AAAAAA`, `BBBBBB`, `CCCCCC`, és `DDDDDD`, és kezdetben 7 opció van. Ezt foglalja össze a következő szavazási konfiguráció (lásd. TXT szintaxisát fentebb).

```
kennedy roosevelt mathias kossuth kádár gandhi mandela
```

Tegyük fel, hogy `AAAAAA` célja, hogy `mathias` nyerjen, `BBBBBB` és `CCCCCC` célja, hogy `mandela` nyerjen, `DDDDDD` célja pedig, hogy `valakimas` nyerjen. Ennek végső soron a gyakorlaton a következő Eclipse-es **Launch Configuration** felel meg (egy már futó JADE platform esetén).

```
java jade.Boot -container -port 1099 -host localhost
AAAAAA:msclab01.votingauction_lab.VoterAgent.VoterAgent(mathias)
BBBBBB:msclab01.votingauction_lab.VoterAgent.VoterAgent(mandela)
CCCCCC:msclab01.votingauction_lab.VoterAgent.VoterAgent(mandela)
DDDDDD:msclab01.votingauction_lab.VoterAgent.VoterAgent(valakimas)
vma:msclab01.votingauction_lab.VotingMechanismAgent.VotingMechanismA
gent(\jade\src\msclab01\votingauction_lab\cfg\voting01.cfg)
```

…az ágensek között ekkor nagyjából a következő kommunikáció zajlik le.

1. `VotingMechanism` → {`AAAAAA`, `BBBBBB`, `CCCCCC`, `DDDDDD`}
   Üzenet típusa: *INFORM*
   Üzenet tartalma: *start*

2. `AAAAAA` → `VotingMechanism`
   Üzenet típusa: *AGREE*
   Üzenet tartalma:

3. `BBBBBB` → `VotingMechanism`
   Üzenet típusa: *AGREE*
   Üzenet tartalma:

4. `DDDDDD` → `VotingMechanism`
   Üzenet típusa: *AGREE*
   Üzenet tartalma:

5. `CCCCCC` → `VotingMechanism`
   Üzenet típusa: *AGREE*
   Üzenet tartalma:

6. `VotingMechanism` → {`AAAAAA`, `BBBBBB`, `CCCCCC`, `DDDDDD`}
   Üzenet típusa: *INFORM*
   Üzenet tartalma: *1 kennedy roosevelt mathias kossuth kádár gandhi mandela*

7. `BBBBBB` → `VotingMechanism`
   Üzenet típusa: *REQUEST*
   Üzenet tartalma: *0 0 0 0 0 0 1*

8. `CCCCCC` → `VotingMechanism`
   Üzenet típusa: *REQUEST*
   Üzenet tartalma: *0 0 0 0 0 0 1*

9. `AAAAAA` → `VotingMechanism`
   Üzenet típusa: *REQUEST*
   Üzenet tartalma: *0 0 1 0 0 0 0*

10. `VotingMechanism` → {`AAAAAA`, `BBBBBB`, `CCCCCC`, `DDDDDD`}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *stop*

A fenti kommunikációból látszik, hogy csak három *Voter* ágens szállt be a gyakorlati szavazásba (7-9 üzenetek). A sima többségi szavazás elvének megfelelően itt már az első körben megszületik a végeredmény, és ezért a szavazás véget ér, hiszen egyetlen opcióra, `mandela`-ra szavaztak a legtöbben. A példában szereplő ágensek haszna az aukció végeztével numerikusan a következő.

$$\text{Utility}(\text{AAAAAA}) = \text{Utility}(\text{DDDDDD}) = 0$$
$$\text{Utility}(\text{BBBBBB}) = \text{Utility}(\text{CCCCCC}) = 1$$

Ezek szerint `BBBBBB` és `CCCCCC` nyert.

Szoftveres jellemzők és követelmények

- A *Voter* ágenseket egy különálló `VoterAgent.java` forrásfájlban találjuk az `msclab01.votingauction_lab.VoterAgent` nevezetű package-ben (a `\jade\src\msclab01\votingauction_lab\VoterAgent` könyvtárban, feltéve, hogy a `\jade\src` benne van a Java classpath-ban).

- A *Voter* ágensek ne rendelkezzenek grafikus felülettel (GUI-val). Információt (minimális mértékben) legfeljebb csak a konzolra, azaz pl. a parancssori ablakba írjanak ki. Legyenek teljesen autonómok, azaz indításukat követően emberi beavatkozás nélkül működjenek, mígnem pl. egy RMA (Remote Management Agent) ágensen keresztül valaki le nem állítja őket.

- A *Voter* ágensek **opcionális** indítási paramétere(i) az általuk (vagy felhasználójuk által) óhajtott opció(k) legyen(ek).

- A *Voter* ágensek semmilyen – közvetlen, vagy közvetett – formában nem hivatkozhatnak a *VotingMechanism* ágens forráskódjára.

- A *VotingMechanism* ágenseket egy különálló `VotingMechanismAgent.java` forrásfájlban találjuk az `msclab01.votingauction_lab.VotingMechanismAgent` nevezetű package-ben (a `\jade\src\msclab01\votingauction_lab\VotingMechanismAgent` könyvtárban).

- Az *VotingMechanism* ágensek rendelkezzenek GUI-val: `VotingMechanismAgentGui.java` forrásfájl az `msclab01.votingauction_lab.VotingMechanismAgent` nevezetű package-ben.

- A *VotingMechanism* ágensek egyetlen, **kötelező** indítási paramétere a szavazási konfigurációs fájl elérése (pl. `\jade\src\msclab01\votingauction_lab\cfg\voting01.cfg`) legyen.

Aukció

A hallgatók feladata a gyakorlat aukciós része során olyan ajánlattevő/licitáló (továbbiakban: ***Bidder***) ágensek létrehozása, melyek képesek különböző aukciókban részt venni. Az aukciókat egy árverező (továbbiakban: ***Auctioneer***) ágens vezeti. A hallgatók számára kezdetben egy angol aukcióra alkalmas *Bidder* és *Auctioneer* ágenst bocsájtunk. A gyakorlat során a hallgatóknak az ágensek alkalmas átírásával különböző aukciókat (pl. japán, holland, szekvenciális) kell megvalósítaniuk. A *Bidder* ágenseket összevetjük egy-egy árverés végén, így fontos a minél jobb licitálási stratégia kialakítása.

Részletes specifikáció

Két fajta ágens fut tehát a JADE platformon: **(1)** *Auctioneer*, és **(2)** *Bidder*. Mindkettő angol aukcióra készen. A hallgatók feladata ezek átírása lesz.

Egy *Auctioneer* és több *Bidder* ágens egy angol aukcióban vesz tehát részt, amihez adott lesz egy a priori, kölcsönösen ismert (lokálisan hozzáférhető/letölthető) leírás. A leírás tartalmazza majd, hogy mi a *Bidder* ágensek kezdőtőkéje (minden *Bidder*-nek ugyanannyi lesz); milyen árufajták vannak, hány darab áll belőlük rendelkezésre[2], és mi ezek minimális kikiáltási ára. Ezeket az adatokat tehát minden *Bidder* ágens induláskor megkapja majd egy megfelelő szövegfájl (TXT) formájában. A fejlesztéskor tehát fel kell készülni az alábbi formátumú szövegfájl megfelelő beolvasására. Ez az **aukció konfigurációja**:

```
fortune I1 N1 P1 I2 N2 P2 … In Nn Pn
```

Itt tehát 4 féle adat szerepel szóközzel elválasztva:

- **(1)** a *Bidder* ágensek kezdőtőkéje [`fortune`], ami egy zérónál nagyobb egész szám,
- **(2)** az árufajta egyedi megnevezése [`Ii`], ami karakterfüzér,
- **(3)** az árufajtából kezdetben rendelkezésre álló áruk darabszáma [`Ni`], ami zérónál nagyobb egész szám, és
- **(4)** az adott árufajtájú áruk minimális kikiáltási ára [`Pi`], ami szintén zérónál nagyobb egész szám.

Tipikusan néhány árufajtáról lesz csak szó, amikből viszont igen sokat ajánlanak majd árverésre (akár egészen véletlenszerű sorrendben). Egy-egy *Bidder*-nek tehát csak néhány áru megvételére lesz pénze, úgyhogy okosan kell gazdálkodnia.

Minden *Bidder* ágensnek azonos a kezdetben rendelkezésre álló össztőkéje (amit vásárlásra fordíthat). A *Bidder* ágensek célja minél több áru minél alacsonyabb áron történő megszerzése. Ezt fejezi ki az alábbi képlet, pontosabban **jósági mérce**.[3]

$$\mathrm{Utility}(\mathrm{Bidder}) = \sum_{x \in \mathrm{Bought\_goods}} \frac{\mathrm{inital\_price\_of}(x)+1}{\mathrm{price\_paid\_for}(x)}$$

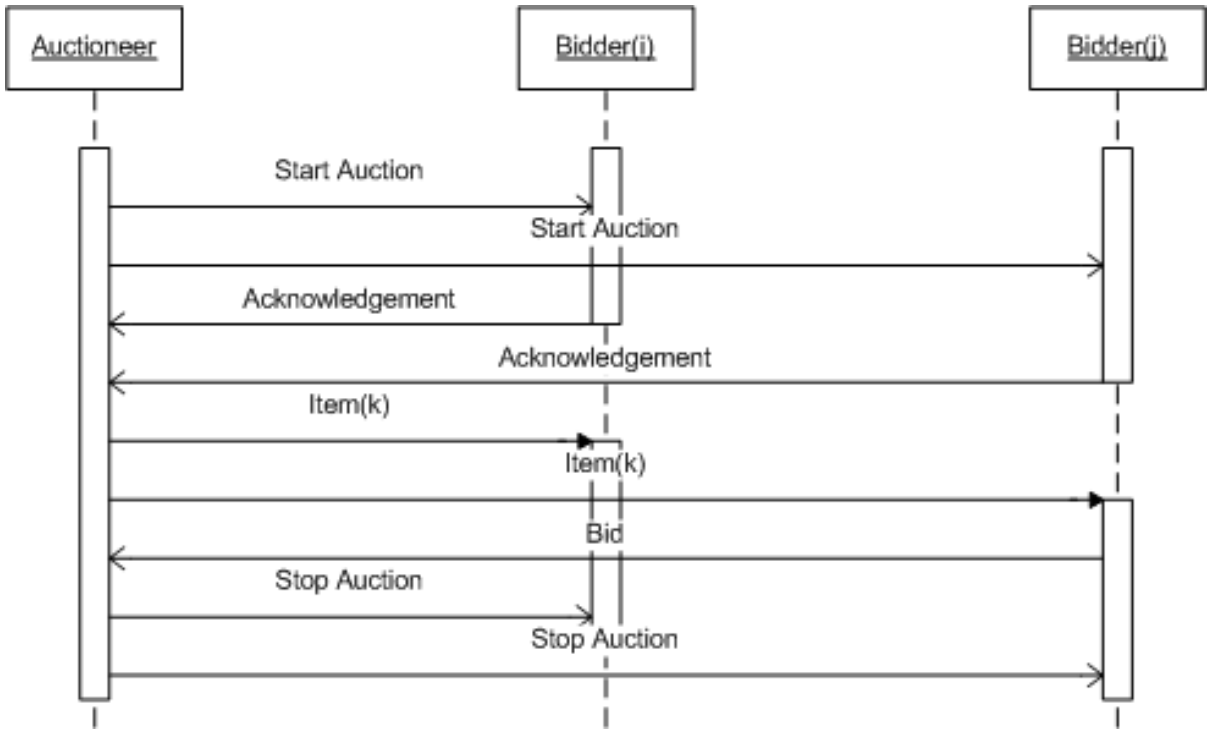
[2] Az aukció során minden áru árverésre lesz bocsájtva.
[3] Az alábbi függvény közvetve arra is ösztönzi az ágenseket, hogy minél több pénzt költsenek, ámde hatékonyan.

Az ágensek célja tehát ennek a függvénynek a maximalizálása. *Megjegyzés: a képletben a számlálóban azért szerepel „+1”, mert így adódik ki az a legkisebb ár, amiért elvben már el lehetne vinni az adott árut.*

Az aukció menete (részletesen)

Mindként fajtájú ágens indulás után azonnal beregisztrál a DF ágensnél. A `ServiceDescription`-ben az *Auctioneer* „`auctioneer`”-t, a *Bidder* pedig „`bidder`”-t ad meg típusként. A szolgáltatás neve is ugyanez legyen mindkettőnél.

A platformon először a *Bidder* ágensek lesznek elindítva, majd ezt követően az *Auctioneer* ágens, amely indulás után lekérdezi a DF-től a platformon aktuálisan jelen lévő, regisztrált *Bidder* ágensek listáját. Ezek után lényegében a következő üzenetváltás zajlik le majd le.



Itt lényegében arról van szó, hogy:

1. Az ***Auctioneer*** kihirdeti, hogy elindult az aukció.
    - Üzenet típusa: *INFORM*
    - Üzenet tartalma: *start*
        iii. ahol a „*start*” egy sima karakterfüzér (és azt jelzi, hogy elindult az aukció)
2. Erre a ***Bidder***-ek egy beleegyezéssel válaszolnak.
    - Üzenet típusa: *AGREE*
    - Üzenet tartalma:
3. Erre az ***Auctioneer*** elkezdi sorban értékesíteni az árukat (egy-egy árufajtából akár többször többet is, véletlen sorrendben). A k-adik értékesített áru esetében először egy olyan broadcast üzenetet küld a *Bidder*-ek számára, amiben jelzi, hogy a k-adik áruról van szó, mi ennek a fajtája, mi az ára (az adott árufajtájú áru kiindulási árának

minimuma az induláskor kapott szövegfájlban szerepel), és hogy hanyadik alkalommal kínálja fel a megadott áron (kezdetben ez nyilván 1). Ez történik először, majd – adott idő elteltével (pl. 5 mp) – másodszor, és harmadszor is. Ha senki sem jelentkezik az áruért, akkor a (k+1)-edik árura kerül a sor (és a k-adik áru nem kerül értékesítésre). Ha viszont valamelyik *Bidder* időközben jelentkezik az áruért egy, a megadottnál magasabb összeg megadásával (lásd. alább a PROPOSE üzenetet), akkor ez az ár beleépül egy a jelenlegihez hasonló, újabb broadcast üzenetbe – először, másodszor, és harmadszor is. Ha senki más sem kínál többet az adott áruért, akkor a legutolsó, legmagasabb ajánlatot tevő *Bidder* ágens lesz a győztes, az övé lesz az árú, és a tőkéje a legutolsó ajánlatában szereplő összeggel csökken.[4] Ezek után a következő árura kerül a sor, avagy ha már nincs több eladni való áru, akkor az egész aukció lezárul, és az *Auctioneer* kiírja az ágenseket minősítő ranglistát.

- Üzenet típusa: *INFORM*
- Üzenet tartalma: *k item price i (bidder)*
    i. ahol a „*k*” egy zérónál nagyobb egész szám (és az aktuálisan kikiáltott áru sorszámát jelzi – kezdetben ez nyilván 1)
    ii. az „*item*” egy karakterfüzér (és azt jelzi, hogy milyen árufajtájú áruról van szó)
    iii. a „*price*” egy zérónál nagyobb egész szám (és vagy az áru kihirdetési árát jelzi (ha nincs megadva „*bidder”* az üzenetben), vagy azt, hogy legutóbb mennyit kínáltak érte (ekkor nyilván ott lesz az üzenet végén az, hogy melyik „*bidder”* tette azt az ajánlatot). FONTOS: a „*price*”-nál mindig **magasabb** árat kell majd kínálnia az ajánlattevőnek (minden egyéb ajánlat érvénytelennek minősül, és nem kerül beszámításra). Nyilván az *Auctioneer*-hez időben elsőként beérkezett ajánlat kerül majd először beszámításra. Ha az *Auctioneer* úgy véli, hogy az adott *Bidder*-nek nincs (már) elég pénze ahhoz, hogy állja az ajánlatát, úgy az ajánlatot semmisnek tekinti.
    iv. az „*i*” értéke 1, 2, vagy 3 lehet attól függően, hogy az adott árut a megadott áron hányadszor bocsájtja árverésre az *Auctioneer*. Ha 3 után adott ideig senki sem jelentkezik, akkor az áru eladásra kerül (vagy raktárba, ha egyáltalán senki sem jelentkezett érte). Nyilván az „*i*” értéke minden egyes újabb érvényes árajánlat után visszaáll 1-re.
    v. a „*bidder*” egy opcionális eleme ennek az üzenetnek, egy karakterfüzér (egy ágens azonosító név, amit az `Agent.getAID().getName()` metódus ad vissza). Ha adott, akkor azt jelzi, hogy ez az ágens kívánja a megadott áron megvenni a megadott árut. Ha nincs megadva, akkor pedig az áru első kihirdetéséről van szó (mikor még senkinek sem volt lehetősége ajánlatot tenni rá).

4. Egy ***Bidder*** ágens, amennyiben egy áru kihirdetéséről (vagy egyik másik *Bidder* ágens elfogadott ajánlatáról) értesül, ajánlatot tehet. Erre szolgál ez az üzenet.
    - Üzenet típusa: *PROPOSE*
    - Üzenet tartalma: *moneybid*
        i. ahol a „*moneybid*” egy zérónál nagyobb egész szám (és azt jelzi, hogy a *Bidder* mennyit kínál az aktuálisan aukcióba bocsájtott áruért). Nyilván

[4] Ez a tény az *Auctioneer* tudásbázisába is beépül. Az *Auctioneer* végig nyomon fogja követni az összes aukcióban résztvevő *Bidder* ágenst. Nyilván fogja tartani az aktuális tőkéjüket, milyen árukat vettek, mennyiért, stb. Ezek az információk az aukció szabályszerű levezetéséhez (pl. ahhoz, hogy a *Bidder* ágensek ne költhessenek többet annál, mint amennyi pénzük van), és az aukció végeztével a *Bidder* ágensek megítéléséhez szükségesek.

csak akkor van esély arra, hogy el legyen fogadva az ajánlat, ha a benne szereplő ár magasabb az aktuálisan adottnál.

5. Amennyiben minden termék aukcióra lett bocsájtva, az ***Auctioneer*** ezzel az üzenettel jelzi, hogy lezárja az aukciót.
    - Üzenet típusa: *INFORM*
    - Üzenet tartalma: *stop*
        iv. ahol a „*stop*” egy sima karakterfüzér (és azt jelzi, hogy lezárult az aukció).

Miután lezajlott az aukció, az *Auctioneer* ágens jelzi (a tárolt és kiszámított adatok alapján), hogy a fenti jósági/hasznossági/sikerességi mércének megfelelően mi a *Bidder* ágensek sikeresség szerint csökkenő sorrendje, és konkrét haszna.

Példa

Tegyük fel a példa kedvéért, hogy két *Bidder* ágens verseng: `AAAAAA`, és `BBBBBB`. Kezdőtőkéjük fejenként 10. Egy fajta áru van csak: `asztal`, amiből kezdetben 3 darab áll rendelkezésre legalább 4 egységnyi kikiáltási áron. Ezt foglalja össze a következő leírás (lásd. TXT szintaxisát fentebb).

```
10 asztal 3 4
```

Ennek megfelelően tegyük fel, hogy az ágensek között a következő kommunikáció zajlik le (a JADE platformra történő belépést, az előbb megadottaknak megfelelő szövegfájl felolvasását, a *DF*-nél történő regisztrációt, és az *Auctioneer DF*-es lekérdezését, és követően).

11. `Auctioneer` → {`AAAAAA`, `BBBBBB`}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *start*

12. `AAAAAA` → `Auctioneer`
    Üzenet típusa: *AGREE*
    Üzenet tartalma:

13. `BBBBBB` → `Auctioneer`
    Üzenet típusa: *AGREE*
    Üzenet tartalma:

14. `Auctioneer` → {`AAAAAA`, `BBBBBB`}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *1 asztal 4 1*

15. `BBBBBB` → `Auctioneer`
    Üzenet típusa: *PROPOSE*
    Üzenet tartalma: *5*

16. `AAAAAA` → `Auctioneer`
    Üzenet típusa: *PROPOSE*
    Üzenet tartalma: *5*

17. Auctioneer → {AAAAAA, BBBBBB}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *1 asztal 5 1 BBBBBB*

18. Auctioneer → {AAAAAA, BBBBBB}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *1 asztal 5 2 BBBBBB*

19. Auctioneer → {AAAAAA, BBBBBB}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *1 asztal 5 3 BBBBBB*

20. AAAAAA → Auctioneer
    Üzenet típusa: *PROPOSE*
    Üzenet tartalma: *6*

21. Auctioneer → {AAAAAA, BBBBBB}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *1 asztal 6 1 AAAAAA*

22. Auctioneer → {AAAAAA, BBBBBB}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *1 asztal 6 2 AAAAAA*

23. Auctioneer → {AAAAAA, BBBBBB}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *1 asztal 6 3 AAAAAA*

24. Auctioneer → {AAAAAA, BBBBBB}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *2 asztal 4 1*

25. Auctioneer → {AAAAAA, BBBBBB}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *2 asztal 4 2*

26. BBBBBB → Auctioneer
    Üzenet típusa: *PROPOSE*
    Üzenet tartalma: *5*

27. Auctioneer → {AAAAAA, BBBBBB}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *2 asztal 5 1 BBBBBB*

28. Auctioneer → {AAAAAA, BBBBBB}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *2 asztal 5 2 BBBBBB*

29. `Auctioneer` → {`AAAAAA`, `BBBBBB`}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *2 asztal 5 3 BBBBBB*

30. `Auctioneer` → {`AAAAAA`, `BBBBBB`}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *3 asztal 4 1*

31. `BBBBBB` → `Auctioneer`
    Üzenet típusa: *PROPOSE*
    Üzenet tartalma: *5*

32. `Auctioneer` → {`AAAAAA`, `BBBBBB`}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *3 asztal 5 1 BBBBBB*

33. `Auctioneer` → {`AAAAAA`, `BBBBBB`}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *3 asztal 5 2 BBBBBB*

34. `Auctioneer` → {`AAAAAA`, `BBBBBB`}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *3 asztal 5 3 BBBBBB*

35. `Auctioneer` → {`AAAAAA`, `BBBBBB`}
    Üzenet típusa: *INFORM*
    Üzenet tartalma: *stop*

A fenti kommunikációból látszik, hogy mindhárom asztal eladásra került. `BBBBBB`-nek szerencséje volt, ugyanis bár az első asztalra `AAAAAA`-val logikailag ugyanakkor licitáltak, fizikailag mégis `BBBBBB` volt a gyorsabb, az ő üzenete ért előbb oda az *Auctioneer*-hez (az üzenetküldés aszinkronitása folytán[5]), és ebből következőleg ő vette át az adott áruval kapcsolatos aukció vezetését (lásd. 7-es üzenet). Ezek után azonban `AAAAAA` mohó volt, és emelte a licitet 6-ra. `BBBBBB` ezt nem tartotta, mert tudta, hogy mindkettőjüknek alapból 10 egységnyi pénze van, és mivel `AAAAAA` immár kénytelen lesz 6-ért megvenni az elsőként felkínált `asztal`-t, ezért a másik két `asztal` már az övé (`BBBBBB`-é) lesz darabonként 5 egységnyi pénzért. Mint látjuk, így is történt: `BBBBBB` hagyta, hogy `AAAAAA` elvigye az első `asztal`-t 6-ért, és utána `AAAAAA` már nem tudott érvényeset licitálni a maradék 2 `asztal`-ra, így aztán `BBBBBB` kikiáltási áron elvitte őket.

A példában szereplő ágensek haszna az aukció végeztével numerikusan a következő:

$$\text{Utility}(\text{AAAAAA}) = \frac{4+1}{6} = \frac{5}{6}$$

[5] Célszerű tehát minél gyorsabb *Bidder* ágenst írni. Mindazonáltal az ágensnek célszerű minél intelligensebbnek lenniük. E két „ellentétes” szempont között kell megtalálni a megfelelő egyensúlyt/kompromisszumot. Ha a *Bidder* ágens intelligens, ámde e miatt lassú, úgy a többiek lekörözhetik. Viszont, ha gyors, ámde kevésbé intelligens, úgy az intelligensebb *Bidder*-ek túljárhatnak az eszén.

$$\text{Utility}(\text{BBBBBB}) = \frac{4+1}{5} + \frac{4+1}{5} = 2$$

Ezek szerint, mivel *2 > 5/6*, ezért `BBBBBB` nyert.[6]

Szoftveres követelmények

- A *Bidder* ágenseket egy különálló `BidderAgent.java` forrásfájlban találjuk az `msclab01.votingauction_lab.BidderAgent` nevezetű package-ben (a `\jade\src\msclab01\votingauction_lab\BidderAgent` könyvtárban, feltéve, hogy a `\jade\src` benne van a Java classpath-ban).

- A *Bidder* ágensek ne rendelkezzenek grafikus felülettel (GUI-val). Információt (minimális mértékben) legfeljebb csak a konzolra, azaz pl. a parancssori ablakba írhatnak ki. Legyenek teljesen autonómok, azaz indításukat követően emberi beavatkozás nélkül működjenek, mígnem pl. egy RMA (Remote Management Agent) ágensen keresztül valaki le nem állítja őket.

- A *Bidder* ágensek egyetlen, **kötelező** indítási paramétere az aukciós konfigurációs fájl elérése (pl. `\jade\src\msclab01\votingauction_lab\cfg\auction01.cfg`).

- A *Bidder* ágensek semmilyen (közvetlen, vagy közvetett formában nem hivatkozhatnak az *Auctioneer* ágens forráskódjára.

- Az *Auctioneer* ágenseket egy különálló `AuctioneerAgent.java` forrásfájlban találjuk az `msclab01.votingauction_lab.AuctioneerAgent` nevezetű package-ben (a `\jade\src\msclab01\votingauction_lab\AuctioneerAgent` könyvtárban).

- Az *Auctioneer* ágensek rendelkezzenek GUI-val: `AuctioneerAgentGui.java` forrásfájl az `msclab01.votingauction_lab.AuctioneerAgent` nevezetű package-ben.

- Az *Auctioneer* ágensek egyetlen, **kötelező** indítási paramétere az aukciós konfigurációs fájl elérése (pl. `\jade\src\msclab01\votingauction_lab\cfg\auction01.cfg`).

[6] A hallgató célja tehát: minél jobb/hasznosabb/sikeresebb *Bidder* ágenst írni! Ha két *Bidder* ágensnek esetleg ugyanakkorára adódna a haszna, úgy rangsorolásukban további szempontként az általuk megvásárolt termékek számát vesszük figyelembe. Ha ez is egyezik mindkét *Bidder* esetében, úgy azonosan jónak tekintjük őket.