



Felhasználói kézikönyv

Készítette:

*Mészáros Róbert, Bocsi Rajmund,
Iványi Márton Dávid, Szabó Attila és
Erdélyi Viktor*



*A projekt az EU társfinanszírozásával, az Európa Terv keretében
valósul meg.*

Budapest, 2007. november

Tartalom

Bevezető	4
Ágens-alapú modellezés	4
MASS	4
Model Exploration Module	5
1 Tudnivalók	6
1.1 Telepítés	6
1.2 Rendszerkövetelmények	6
1.2.1 Hardver	6
1.2.2 Szoftver	6
2 Panelek (a Window menü)	7
2.1 Megjelenítés	7
2.2 Színsémák (Skins), témák (themes), vízjelek (watermarks)	8
2.3 Helyi menük	8
3 Szimulációs eredmények tárolása és rendszerezése	9
3.1 Konceptió: Modell név és verzió	9
3.2 Konceptió: Input és output paraméterek	9
3.3 Fájl- és adatbázisbeállítások	9
3.4 A futás közben használt memória (heap) mérete	11
3.5 Fájlimportálás	11
3.5.1 Repast importálás	11
3.5.2 CSV importálás	13
3.6 Fájllexportálás	15
3.7 Az eredmények böngészése (Results browser)	16
3.8 Eredmények törlése	17
4 Nézetek	18
4.1 Konceptió: nézet-tábla	18
4.2 Nézetek létrehozása	18
4.2.1 1. lépés – számítások	19
4.2.2 2. lépés - rendezés	20
4.2.3 3. lépés – név és leírás	21
4.2.4 Az eredmény	21
4.2.5 0. lépés – kiindulási adatok	23
4.3 Nézetek törlése	23
4.4 Nézetek szerkesztése (Recreate)	23
5 Grafikonok	25
5.1 Grafikonok létrehozása (Charts/Create chart...)	25
5.2 Konceptió: Adatforrások	26
5.3 A Compose, Display, Save és Cancel gombok	26
5.4 Nagyítás	27
5.5 Ábrák mentése	27
5.6 Tulajdonságok (Properties)	27
5.7 A Details felület	27
5.8 Grafikon megnyitása	28
5.9 Grafikontípusok	28
5.9.1 Oszlopdiagram (Bar Chart)	28

5.9.2	2D rács (2D Grid)	29
5.9.3	Alakzat rács (Shape Grid)	30
5.9.4	Vegyes rács (Composite Grid)	31
5.9.5	Hisztogram	31
5.9.6	Hálózat (Network)	31
5.9.7	Területalapú diagram (Rectangle Area Chart)	31
5.9.8	Pontdiagram (Scatter Plot)	31
5.9.9	Sorozat (sequence)	32
5.9.10	Idősor (Time Series)	32
5.9.11	Vonaldiagram (XY Line Chart)	32
6	Scriptelés	33
6.1	Nézet scriptjének mentése	33
6.2	Nézet scriptjének betöltése	33
6.3	Nézet-táblák szerkesztése	33
6.3.1	Példakód	33
6.3.2	Megjegyzések	34
6.4	Beanshell scriptelés	34
6.4.1	Beépített műveletek	34
7	Ismert problémák (Known issues)	36
7.1	MEME Parameter Sweep Tool és Monitor	36
8	Hibaelhárítás	37
8.1	Hibajelentések	37
8.2	Memória használat	37
8.3	Help menü	37
8.4	A művelet állását jelző ablak (progress window)	37
9	Zárszó	38
10	Referenciák	39

Bevezető

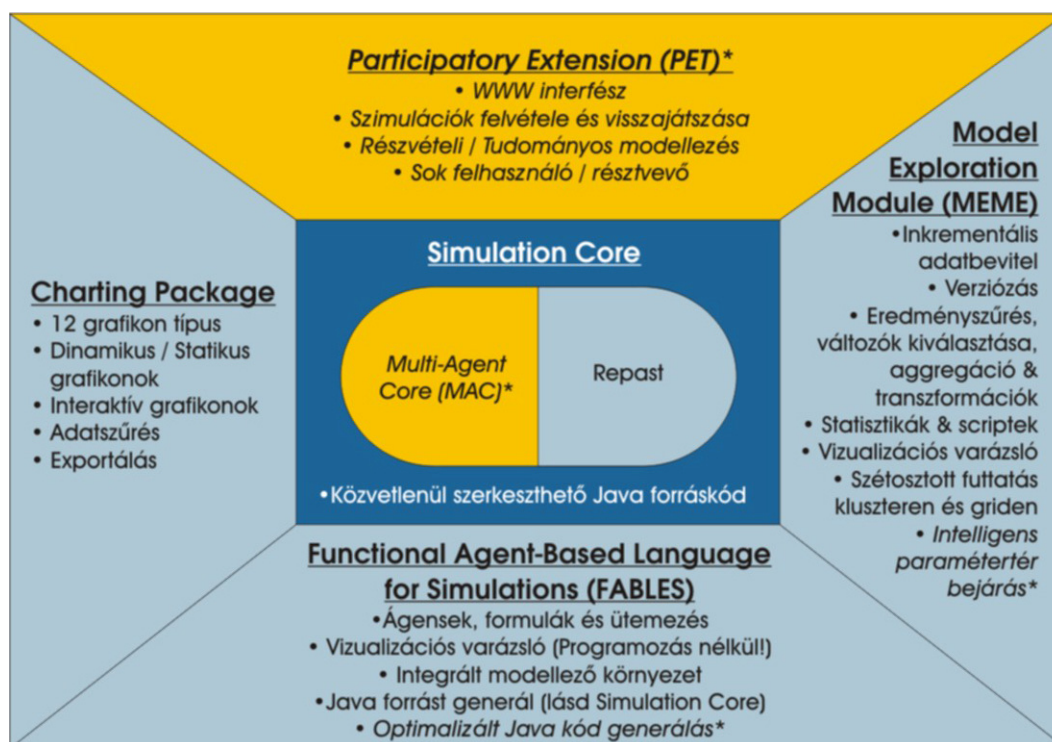
Ágens-alapú modellezés

Az ágens-alapú modellezés a számítógépes szimulációk egy - komplex társadalmi rendszerek modellezésére különösen alkalmas - új ága. Alapvetése az, hogy az egyént modellezzük, tökéletlenségeivel (pl. korlátozott kognitív és számítási képességek), egyéni jellegzetességeivel és egyedi interakcióival együtt. A modell tehát "alulról felfelé" épül - elsősorban a mikro szabályokra koncentrálva, de a makró jelenségek kialakulását kutatva. A részvételi szimuláció, mint az ágens-alapú szimuláció alfaja egy olyan metodológia, amely az emberi szereplők és a mesterséges ágensek együttműködésére alapoz. Ezek a megoldások a képzési és döntés-támogatási területeken igen hasznosak.

MASS

A Multi-Agent Simulation Suite (MASS) egy szoftvercsomag, amely lehetővé teszi a felhasználó számára, hogy az ágens-alapú modellezés megoldásait változatos területeken alkalmazza anélkül, hogy komoly programozási ismereteket kelljen elsajátítania.

A programcsomag négy, egy ún. szimulációs mag köré szerveződő alkalmazásból áll össze. A MASS rendelkezik egy saját maggal (ez a MAC), illetve képes futni Repast alapokon is. A több szimulációs magon való futtathatóság biztosítja, hogy a modellek mag-függetlenek legyenek, így a használható magok számát a jövőben bővíteni kívánjuk. A Functional Agent-Based Language for Simulation (FABLES) egy olyan programozási nyelv és modellezési környezet, amely kifejezetten ágens-alapú modellek fejlesztését szolgálja. A Model Exploration Module (MEME) paraméter terek bejárását, a kinyert adatok feldolgozását és megjelenítését hivatott támogatni. A Participatory Extension (PET) egy opcionális web-alapú környezet multi-ágens és részvételi szimulációk futtatásához. A MASS negyedik része, a Vizualizációs Csomag, nem jelenik meg önálló programként, a többi szoftverben használt grafikonok és vizualizációk implementációit tartalmazza.



1. ábra - Multi-Agent Simulation Suite

Model Exploration Module

Ön most a MASS Model Exploration Module (MEME) felhasználói kézikönyvét olvassa. A modell szó ebben a kontextusban bármely, MASS-ban vagy Repast-ban íródott ágens alapú szimulációs programot jelent, amely megkísérel szimulálni, mi több *modellezni* egy adott jelenséget a valós életből. Az ilyen szimulációkról általánosságban elmondható, hogy egy adott modell működésének tanulmányozását – mondhatjuk, hogy felfedezését – teszi lehetővé. A modellek ilyenén való feltérképezéséhez számos faktort kell egyszerre megfigyelni a szimulációk futása során (illetve elemezni a futások után), mely futások sok esetben igen nagy számoságúak, illetve hosszúak (amit számítógépes szaknyelven batch futtatásnak hívnak). Ezen megfigyelések során nagy mennyiségű elemzendő nyers adat keletkezik.

A MEME egy eszköz az ilyen batch futtatások, illetve az azok által eredményezett adatok kezelésére. Lehetővé teszi a felhasználó számára, hogy adatbázis(ok)ban tárolja és rendszerezze a nyers adatokat, és hogy létrehozson olyan letisztított táblákat, amelyek már megjeleníthetők grafikonokon, diagramokon. A közeljövőben a MEME-t képessé tesszük a MASS/Repast szimulációk közvetlen futtatására és az eredmények belső gyűjtésére. A szimulációk helyi számítógépen, illetve griden és számítógép klaszteren történő futtatását a program jelenleg még különálló, MEME Parameter Sweep Tool modulja valósítja meg. A fejlesztés következő részében ezt az eszközt is integráljuk a programba.

Terveink szerint ugyancsak lehetővé tesszük, hogy olyan kifinomult statisztikai szoftverekkel működjön együtt, mint a Matlab, vagy az R. Ezek a fejlesztések tovább fogják egyszerűsíteni a modellezés folyamatát, aminek eredményeképp a felhasználó még inkább magukra a modellekre koncentrálhat.

1 Tudnivalók

1.1 Telepítés

A MEME egy öntelepítő exe fájl formájában szerezhető be, illetve tölthető le (<http://meme.aitia.ai/>). A telepítő varázsló végigvezeti a felhasználót a telepítésen, melynek során ellenőrzi az esetleg már telepített MEME verzióját és a szükséges Java Runtime Environment meglétét (1.5 vagy későbbi, ezt is telepíti, amennyiben szükséges), felajánlja a Repast J program telepítését, és bekéri a program telepítése szempontjából szükséges beállításokat (telepítési könyvtár, Start Menü beállítások, stb.).

A telepítő a megadott célkönyvtárba másolja a MEME-t, a megfelelő helyeken elhelyezi a szükséges parancsikonokat, és bejegyzi a kurrens verziószámot a registry-be a telepítés során. A futáshoz szükséges adatbázisfájlok és egyéb leíró fájlok az első futtatás alkalmával jönnek létre.

1.2 Rendszerkövetelmények

1.2.1 Hardver

Minimum követelmények: 1 GHz CPU, 512 MB RAM

Javasolt: Dual-core CPU, 1 GB RAM

1.2.2 Szoftver

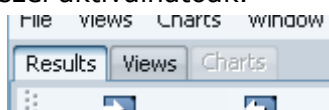
Bármely rendszer, amelyen lehet futtatni a Java JRE 1.5-öt, az a MEME-t is képes lesz futtatni. Az egyes operációs rendszerekre vonatkozó részletes követelmények az alábbi címen érhetőek el:

<http://java.com/en/download/help/5000011000.xml>

A MEME bármely Java futtatására alkalmas operációs rendszeren működik, habár Mac-en vagy Linuxon való tesztelése még várat magára.

2 Panelek (a Window menü)

A menüsor alatt, a MEME ablak bal oldalán fülek találhatóak. Ezekre a fülekre kattintva a felhasználói felület különböző részei aktiválhatóak.

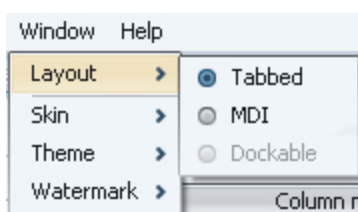


2. ábra - Panelek

Az első – *Results* – az eredményböngészőt (*Results browser*) jelenti meg (lásd 3.7 Az eredmények böngészése (*Results browser*) részt!), ahol a szimulációs eredményekkel lehet dolgozni. A második a *Views* elnevezésre hallgat, itt nézetablakkal foglalkozhatunk, amelyekről bővebben a 4 Nézetek részben értekezünk. A *Charts* oldal csak akkor elérhető, amikor egy grafikon megjelenítésén dolgozunk.

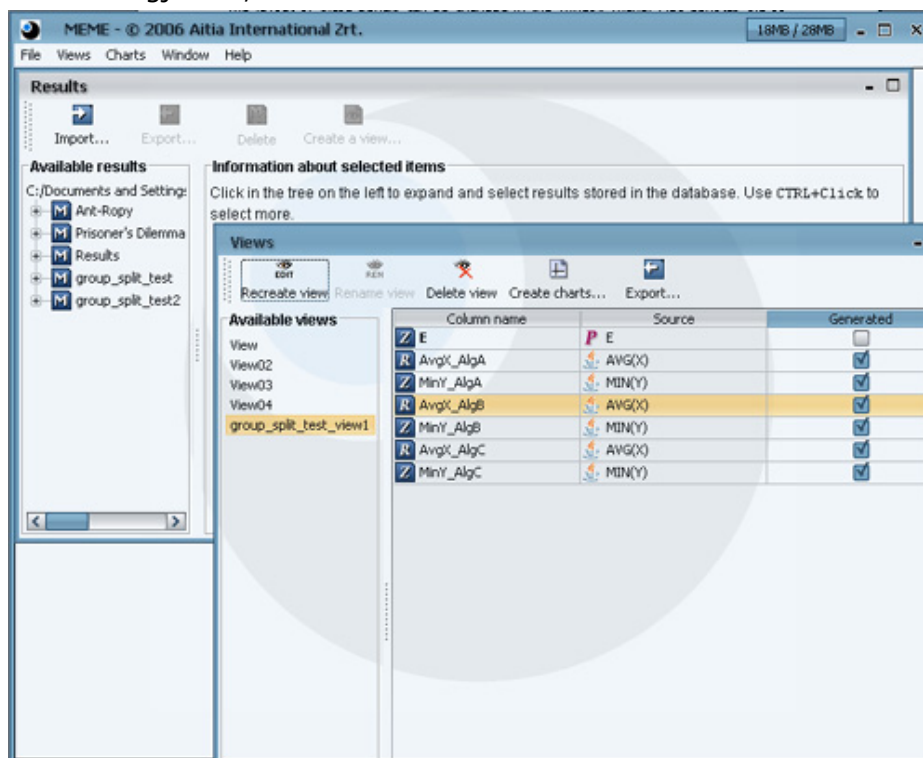
2.1 Megjelenítés

Ezen oldalak, panelek megjelenítése változtatható a *Window > Layout* menüponton keresztül.



3. ábra - Ablak típusok

A „*MDI*” az úgynevezett „*multi document interface*”-t jelöli, míg „*Tabbed*” az alapbeállításként megjelenő, füles ablakokat.



4. ábra - MDI

2.2 Színsémák (Skins), témák (themes), vízjelek (watermarks)

A színséma, téma vagy vízjel megváltoztatható a *Window* menü megfelelő menüpontjában.

Megjegyzendő, hogy a beállítások hatása csak újraindítás után lesz látható!

2.3 Helyi menük

Táblákra, listákra, és fastruktúrákra jobb egérgombbal kattintva helyi menük érhetők el, amelyek kontextustól függő parancsokat tartalmaznak. Példaként lásd 18. ábra - Nézetek!

3 Szimulációs eredmények tárolása és rendszerezése

A továbbiakban a szimuláció eredménye alatt egy adag nyers adatot értünk, melyek egy szimuláció egyszeri futása során keletkeztek.

3.1 Konceptió: Modell név és verzió

Ahogy az a 3.7 Az eredmények böngészése (Results browser) részben bemutatásra kerül, a MEME a szimulációs eredményeket egy háromszintes hierarchiában rendezi el. Minden eredmény egy adott modell (model) adott verziójához (version) tartozik. A „model” és a „version” itt felhasználó által megadható tetszőleges nevek¹. Ezek jelentik a hierarchia első két szintjét. A harmadik szint a „kötegeket”, azaz a batch futtatásokat tartalmazza. Általában egyszerre több futtatást csinálnak, amiből szimulációs eredmények egy kötege keletkezik, ezért minden szimulációs eredmény egy adott batch-hez is tartozik. A batch-eket a MEME által automatikusan kiosztott számozás azonosítja.

3.2 Konceptió: Input és output paraméterek

A szimuláció eredményei olyan értékekből állnak melyek, a szimuláció futása alatt vagy után kerülnek rögzítésre. Ezeket az értékeket általában valamilyen névvel azonosítják, itt a továbbiakban *paraméterek*nek nevezzük őket. A MEME megkülönböztet *input* és *output* paramétereket. Az input paraméterek olyan értékeket jelentenek, amelyek azokat a körülményeket reprezentálják/írják le, amelyekben az adott futás során a modell leledzik, ezek tehát egy futás során változatlanok. Az output paraméterek azok a faktorok, amelyek változhatnak a szimuláció futása során, amelyeket megfigyelünk, és eredményként rögzítünk. Három fontos dolgot kell megjegyeznünk a paraméterekkel kapcsolatban:

- Egy adott modell adott verziójához tartozó szimulációs eredmények megegyező paraméterkészlettel rendelkeznek. Olyan eredmények adatbázishoz adását, melyek eltérő paramétereket is tartalmaznak az eredeti adatbázishoz képest, nem ajánljuk (ugyanakkor a program nem tesz ilyen korlátozást: az új paraméterek is belekerülnek a korábbi eredményekbe `null` értékkel, illetve az új eredményekből hiányzó, korábbi paraméterek is bekerülnek az új eredményekbe, ugyancsak `null` értékkel).

Az általános szabály ugyanakkor az, hogy az eltérő paraméterkészletek lehetőleg eltérő verziókban, vagy modellekben kapjanak helyet.

- A paraméter elnevezéseknek kategórián belül különbözőnek kell lenniük, azaz nem lehet két input paraméter „X” néven, de lehet egy input és egy output paraméter is azonos néven. Az elnevezések érzékenyek a kisbetű-nagybetű különbségekre, így például a „pH” és a „Ph” eltérő paramétereket jelölnek. A paraméter nevek tartalmazhatnak speciális karaktereket és szóközöket, de sortörést nem.
- A MEME csak a szám, szöveg és logikai (igaz-hamis) értékeket támogatja.

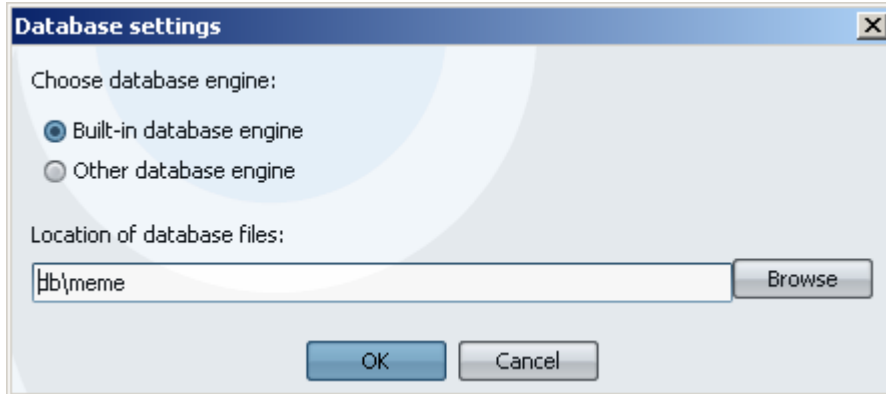
3.3 Fájl- és adatbázisbeállítások

A MEME a szimulációs eredményeket adatbázisban tárolja, amit egy SQL adatbáziszerver kezel, és amelyhez a MEME elindulásakor kapcsolódik. Alapbeállításaként a MEME egy beépített SQL adatbáziszerverrel használ, ami az adatokat a helyi

¹ A modell névnek egyedinek kell lennie, és maximum 64 karakterből állhat. Hasonlóan, a verzió név is egyedi kell, hogy legyen a modellen belül, és ez is csak 64 karakterből állhat.

fájlrendszeren tárolja. Ugyanakkor ez az adatbázismotor nem annyira erőteljes, mint a professzionális DMBS megoldások, és az adatbázis méretének felső határa² 8 GB. Ezen okok miatt a program támogatja a különböző professzionális adatbázis-kezelők JDBC protokollon keresztüli használatát.

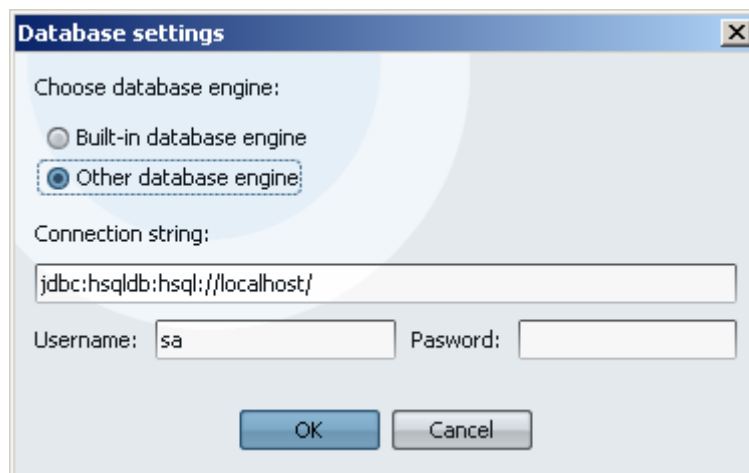
A *Database settings* a *File* menüben lehetőséget biztosít a használni kívánt adatbázis beállítására:



5. ábra – A beépített adatbázismotor beállítása

Ha a beépített adatbázismotort kívánja használni, meg kell adnia egy könyvtárhivatkozást és egy fájlnevet. Ha a hivatkozás nem abszolút, akkor a MEME telepítési könyvtárhoz³ képest relatív módon kerül értelmezésre. A fenti példánál maradván a `db\meme` azt jelenti, hogy az adatbázis a `db` alkönyvtárban, a `meme.backup`, `meme.data`, `meme.properties` és `meme.script` fájlokban kerül eltárolásra.

Ennek alternatívája a külső adatbázisszerver használata. Ebben az esetben meg kell adni egy felhasználó nevet, egy jelszót és egy JDBC elérési utat, ahogy a következő ábrán is látható:



6. ábra – Külső adatbázismotor beállítása

Ezen az ablakon keresztül lecserélheti az éppen használni kívánt adatbázist. A legutolsó beállítást a program megjegyzi, és automatikusan használja a következő indításnál is.

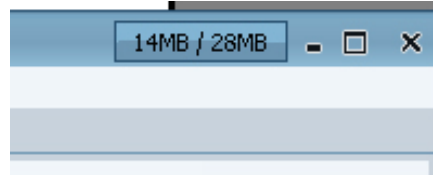
Felhívjuk a figyelmet arra, hogy a MEME nem támogatja az adatok adatbázisok közötti mozgatását. Más szóval élve, ha egy adatot eltárolt egy adatbázisban, azt nem tudja áthelyezni egy másikba a MEME segítségével.

² Ha az adatbázis mérete meghaladná a 8GB-ot, akkor egy hibaüzenet jelenik meg, és az alkalmazás leáll.

³ Alapértelmezésben ez a `C:\Program Files\MASS\MEME` könyvtár.

3.4 A futás közben használt memória (heap) mérete

A MEME ablak jobb felső sarkában elhelyezkedő panelen kerül megjelenítésre a heap mérete, illetve a teljes rendelkezésre álló memória mérete. A MEME alapbeállítása legfeljebb 256 MB-ot használ az adatbázis műveletekhez. Ez a beállítás szükség esetén megváltoztatható (lásd a 8.2 Memória használat részben!). A program bizonyos időközönként elindítja a garbage collector-t a már nem használt memória felszabadítására. Ez a műveletet a panelra kattintva a felhasználó is kezdeményezheti.



7. ábra - Heap panel

3.5 Fájlimportálás

A MEME képes beolvasni szimulációs eredményeket az adatbázisába. Az *Import...* parancs alatt a *File* menüben megtalálható a támogatott formátumok listája. A következő szakaszokban ismertetjük ezen formátumok importálási tulajdonságait.

3.5.1 Repast importálás

A RepastJ lehetővé teszi a szimulációk során az egyes változók megfigyelését, és azok mentését egy log fájlba, melynek formátuma a következőhöz hasonló:

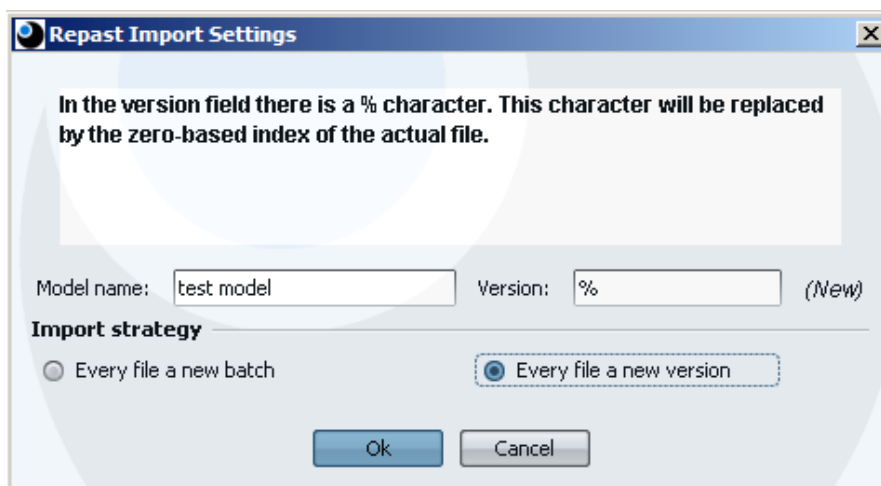
```
Timestamp: 2006.12.13. 10:19:56
Both: 1.0
Looser: -3.0
Neither: 0.0
Payoff1: 0
Payoff2: 0
StopAt: 10000.0
Strat1: 2.0
Strat2: 2.0
Winner: 4.0

"run", "tick", "RngSeed", "payoff1", "payoff2"
1,10000.0,1166001596609,10001.0,10001.0
2,10000.0,1166001596921,10001.0,10001.0
3,10000.0,1166001596953,10001.0,10001.0
4,10000.0,1166001596984,10001.0,10001.0
5,10000.0,1166001597000,10001.0,10001.0

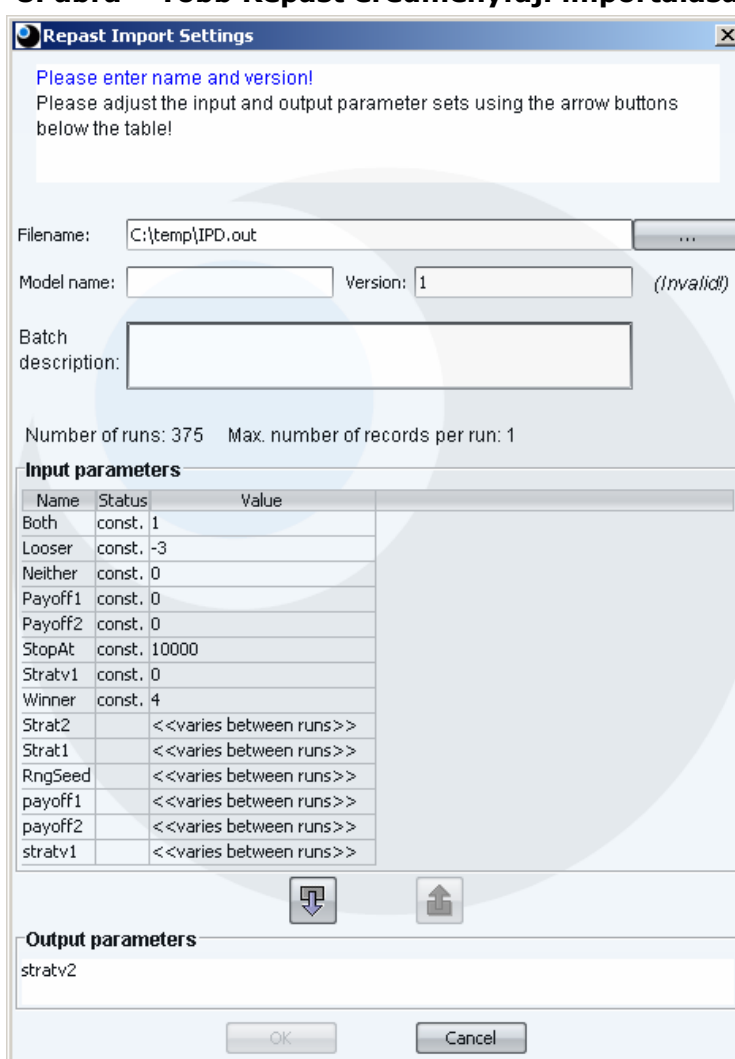
End Time: 2006.12.13. 10:19:57
```

A *File / Import... / Repast result file* menüparancsot használva egy fájl megnyitása párbeszédablakon keresztül lehet kijelölni, és betölteni a kívánt eredményeket hordozó Repast eredményfájlt az éppen használt adatbázisba. Az alkalmazás automatikusan felismeri az elválasztó karaktereket, a paraméterek listáját és azok típusát. Ugyancsak megkísérli felismerni, hogy mely paraméterek input, illetve output jellegűek. E művelet eredményeit egy táblázatban jeleníti meg, ahol a felhasználó pontosíthatja ezeket az információkat. (Lásd 9. ábra!)

Több Repast eredményfájl megnyitása a fájl megnyitása párbeszédablakban, az importálni kívánt fájlok kijelölésével történik (a CTRL billentyű lenyomása mellett a megfelelő fájlokra kattintva lehet több fájlt kijelölni). Ezután meg kell adni a modell nevét és a verziószámot (mindkét esetben szám, illetve szöveg érték adható meg), ha minden fájlt külön batch-ként szeretnénk eltárolni, vagy választható az *Every file a new version* opció. Utóbbi esetben a verziószámítás automatikusan történik (a % helyére kerül a generált szám). Több fájl megnyitása esetén a felhasználó nem állíthatja be az paraméterek input, illetve output jellegét. (Lásd 8. ábra!)



8. ábra – Több Repast eredményfájl importálása



9. ábra – Input és output paraméterek meghatározása

Az *Output parameters* és *Input parameters* mezők között található felfele és lefele nyilak segítségével határozható meg, hogy melyik paraméter melyik csoporthoz tartozik. Legalább egy output paraméternek kell lennie, és azok a paraméterek, amelyek a fájl fejlécében listázva vannak, nem lehetnek output paraméterek.

Ez a párbeszédablak ugyancsak bekéri, hogy az eredmények mely modellhez és verzióhoz tartoznak. Mivel ezek a mezők korlátozott hosszúságúak (maximum 64

karakter), van egy leírás mező is, ahol a modell bővebben bemutatható. Amennyiben a megadott modellnév és/vagy verzió még nem létezik, a program létrehozza azt az adatbázisban. Ha már létezik a megadott néven entitás, akkor az újonnan beolvasott adatok új batch-ként adódnak hozzá a létező modell verzióhoz.

Ha a megadott modell és verzió már létezik, azt jelzi a *Version* mező mellett megjelenő címke. Amennyiben a paraméterek eltérnek a korábbiaktól, az új, illetve a hiányzó paraméterek jelölésre kerülnek a táblázatban és az *Output parameters* listában (lásd 3.2 Konceptió: Input és output paraméterek!). Ha csak a paraméterek típusa tér el, arról nem ad külön jelzést a program, de minden meglévő adat a tágabb típusra konvertálódik. (Például, ha az *X* paraméter már létezett, mint numerikus érték, de az importálásnál *X*, mint szöveges érték kerülne be, az összes korábbi *X* érték is szöveggé konvertálódik.)

Az *OK* gomb csak akkor nyomható meg, ha a modellnév és verzió mezők megfelelően vannak kitöltve.

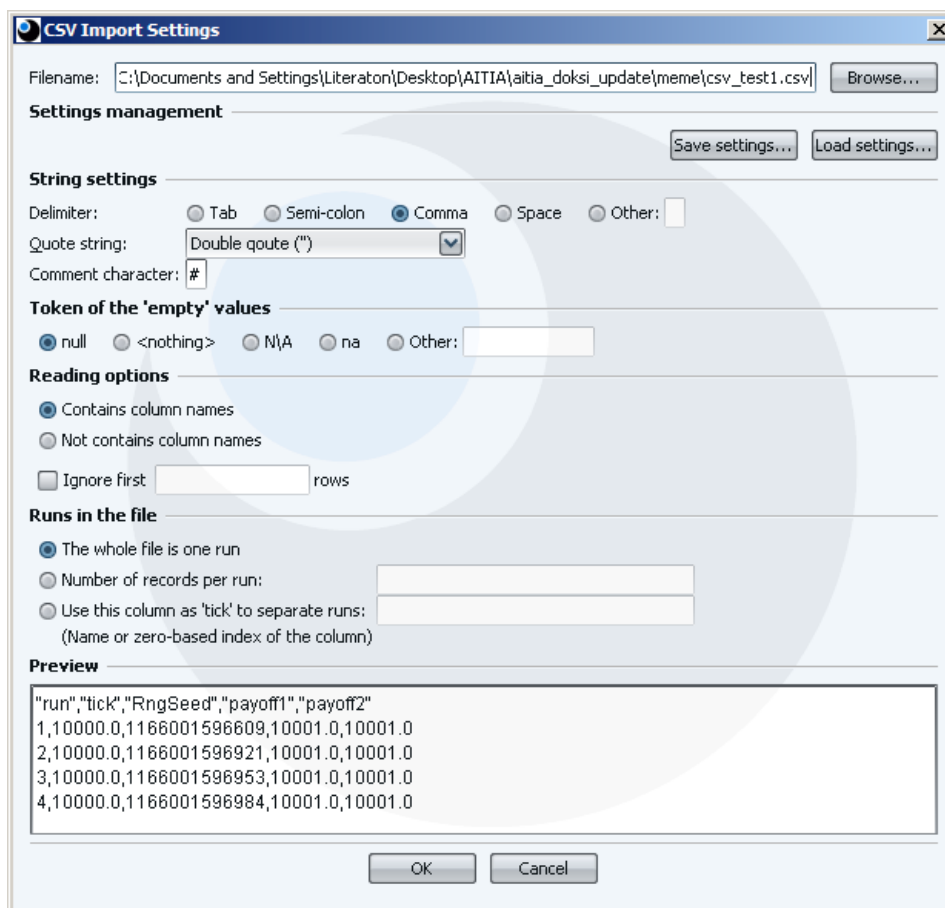
3.5.2 CSV importálás

A MEME CSV (Comma Separated Values – „vesszővel határolt értékek”) fájlokat szintén képes megnyitni. A CSV fájlok formátuma a következőhöz hasonló:

```
"run", "tick", "RngSeed", "payoff1", "payoff2"
1,10000.0,1166001596609,10001.0,10001.0
2,10000.0,1166001596921,10001.0,10001.0
3,10000.0,1166001596953,10001.0,10001.0
4,10000.0,1166001596984,10001.0,10001.0
5,10000.0,1166001597000,10001.0,10001.0
```

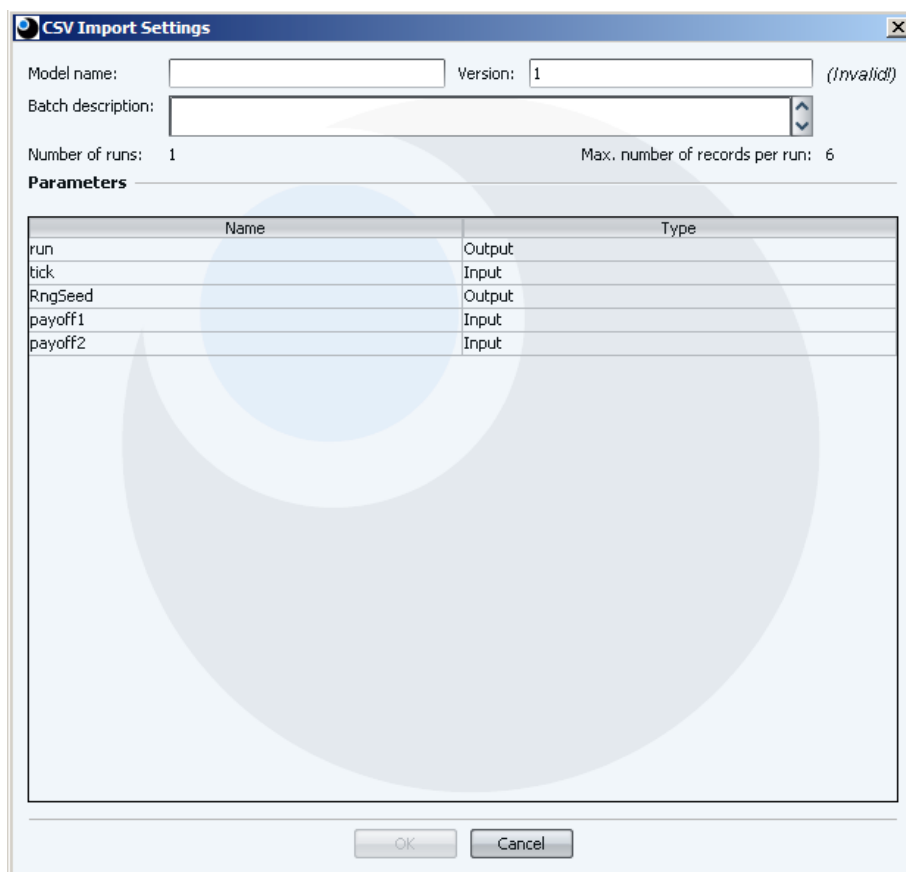
A megfelelő CSV fájl kiválasztása után megjelenik a *CSV Import Settings* párbeszédablak, ahol a következők állíthatók: határoló jel, idézőjel, megjegyzés-karakter, üres értéket reprezentáló jelsorozat, olvasási beállítások (oszlopnevek elhagyása, az első *x* sor elhagyása), futtatási beállítások (a fájl egy futtatást tartalmaz, megadott számú sor számít egy futtatásnak, egy megadott oszlop értékei különítik el a futtatásokat). Az előnézeti mező mutatja a betöltendő adat első néhány sorát a megadott beállítások szerint. (Lásd 10. ábra!)

A legutolsó beállítást a program megjegyzi, és automatikusan használja a következő indításkor is, de lehetőség van a beállítások elmentésére, és visszatöltésére is.



10. ábra – Egy CSV fájl importálási tulajdonságai

Az *OK* gomb megnyomása után a 3.5.1 Repast importálás részben bemutatotthoz hasonló, a modell, verzió és batch beállításokat tartalmazó párbeszédablak jelenik meg (lásd 11. ábra!). Az egyetlen különbség az, ahogy most az *Input* jellegűnek felismert paraméterek típusa *Output* jellegűre változtatható: ehhez a megfelelő paraméter *Type* mezőjére kell kattintani, és a legördülő listából kiválasztani az *Output* típust. Egyszerre több CSV fájl beolvasása hasonlóan végezhető el, mint a Repast eredményfájlok beolvasása.

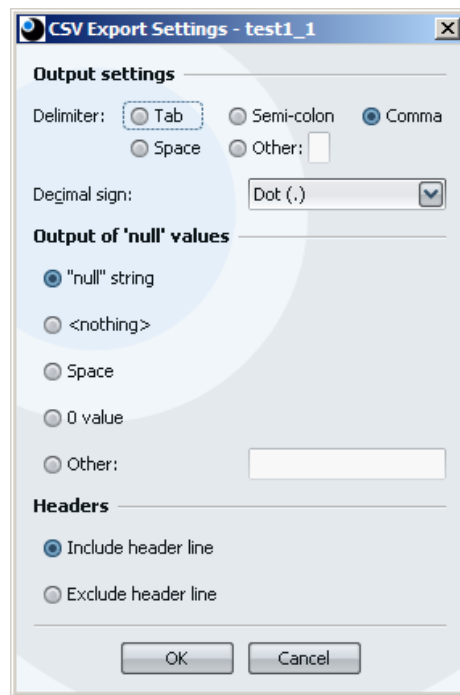


11. ábra – CSV formátumú adatok importálási tulajdonságai

3.6 Fájlexportálás

A nézet-táblák (lásd 4 Nézetek!), a MEME adatbázis rendszerezett adatai, és a verziók (lásd 3.1 Konceptió: Modell név és verzió!) a *File > Export...* paranccsal CSV fájlba exportálhatók. Az exportálás során megadható a határoló jel, tizedes jel, az üres értéket reprezentáló jelsorozat, és beállítható a fejléc (lásd 12. ábra!).

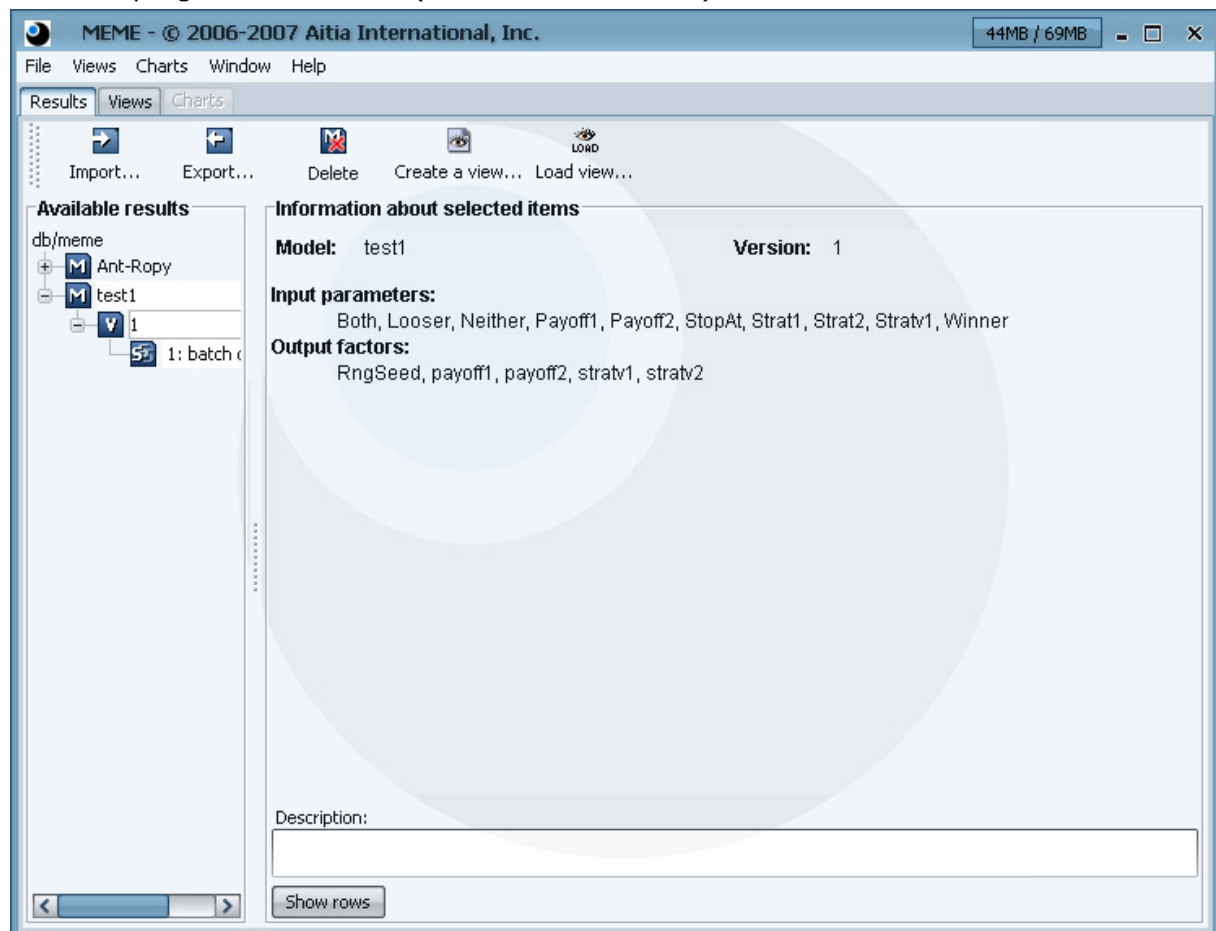
Alapértelmezésként az exportált fájl neve nézet-tábla esetén *<nézet neve>.csv*, verzió esetén *<modell>_<verzió>.csv* lesz. Egyszerre több nézet-tábla vagy verzió is exportálható azonos beállításokkal. Ebben az esetben a fájlok az alapértelmezett módon kerülnek elnevezésre.




12. ábra – CSV exportálási beállítások

3.7 Az eredmények böngészése (Results browser)

Amint a szimuláció eredményei tárolásra kerültek az adatbázisban, azok megtekinthetővé válnak a program főablakból. (Lásd 13. ábra alább!)

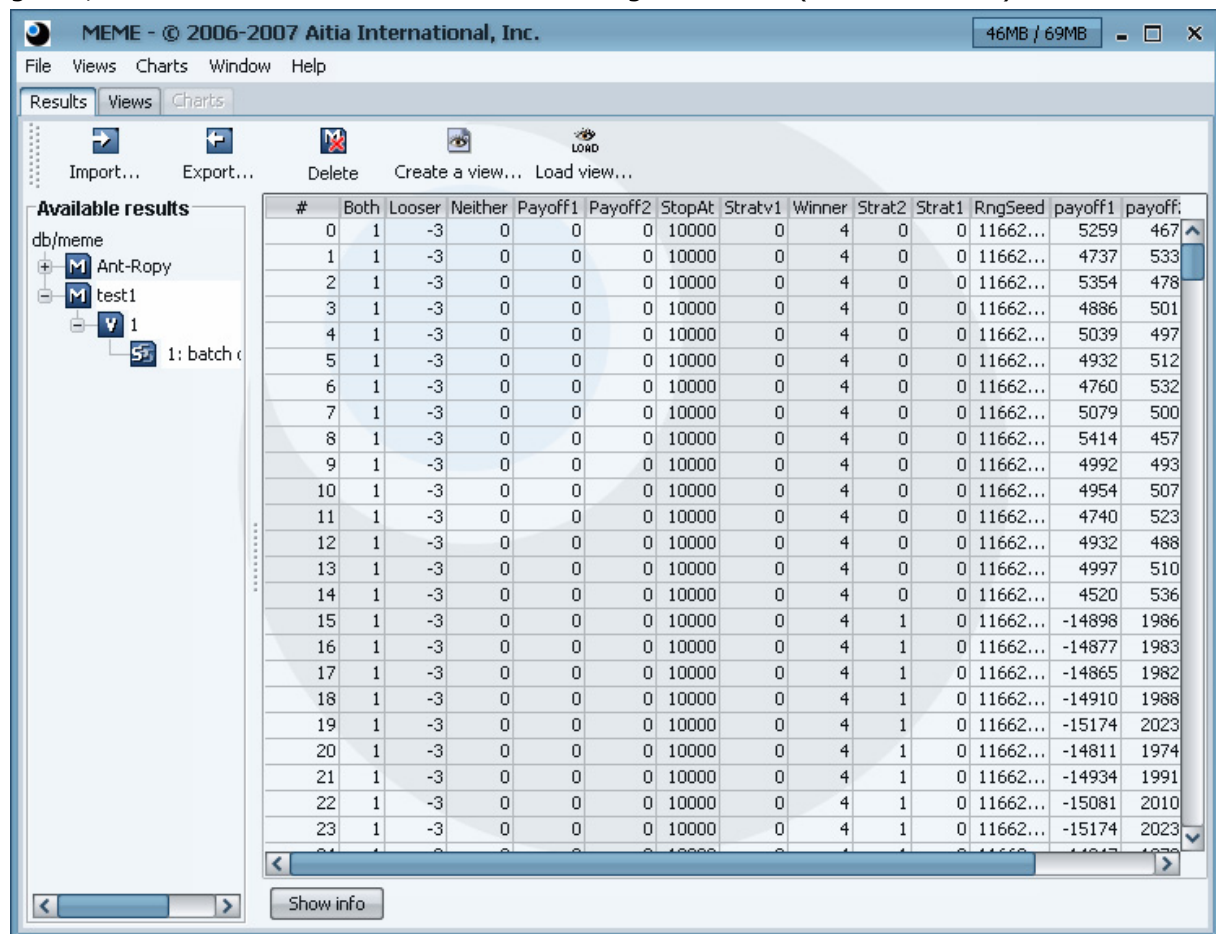


13. ábra - Eredményböngésző

Az eredmények hierarchiája az ablak bal oldalán látható fastruktúrában kerül megjelenítésre. A fa gyökere a használt adatbázis helyét jelöli (db/meme), melyből a különböző modellek nevei ágaznak le. Ezek kibonthatóak dupla kattintással, vagy a modell neve mellett a -ra egyszer kattintva. Egy nevet kiválasztva, részletes információk jelennek meg az adott modellről vagy verzióról a jobb oldalon. A *Description* mező átírható.

Egyszerre több ág is kiválasztható. Általánosságban elmondható, hogy ha egy ág kiválasztunk, akkor annak összes leszármazottja is ki lesz jelölve ezáltal. Például, egy modell kiválasztásánál annak összes verziója és batch-e is belekerül a kijelölésbe. Amikor egyszerre több verzió van kiválasztva, a róluk megjelenített információ azok összes paramétereit tartalmazza. Ha egy modell több verziójából szándékozunk adatokat használni – például azokból nézet-táblákat akarunk létrehozni (lásd a 4.2-es részt!) –, igen hasznos, hogy egyszerre több elem is kiválasztható.

Amikor egy verziót, batch-et, vagy több batch-et kijelölünk, megjelenik a *Show rows* gomb, lehetővé téve a kiválasztott adatok megtekintését. (Lásd 14. ábra!)




#	Both	Loser	Neither	Payoff1	Payoff2	StopAt	Stratv1	Winner	Strat2	Strat1	RngSeed	payoff1	payoff2
0	1	-3	0	0	0	10000	0	4	0	0	11662...	5259	467
1	1	-3	0	0	0	10000	0	4	0	0	11662...	4737	533
2	1	-3	0	0	0	10000	0	4	0	0	11662...	5354	478
3	1	-3	0	0	0	10000	0	4	0	0	11662...	4886	501
4	1	-3	0	0	0	10000	0	4	0	0	11662...	5039	497
5	1	-3	0	0	0	10000	0	4	0	0	11662...	4932	512
6	1	-3	0	0	0	10000	0	4	0	0	11662...	4760	532
7	1	-3	0	0	0	10000	0	4	0	0	11662...	5079	500
8	1	-3	0	0	0	10000	0	4	0	0	11662...	5414	457
9	1	-3	0	0	0	10000	0	4	0	0	11662...	4992	493
10	1	-3	0	0	0	10000	0	4	0	0	11662...	4954	507
11	1	-3	0	0	0	10000	0	4	0	0	11662...	4740	523
12	1	-3	0	0	0	10000	0	4	0	0	11662...	4932	488
13	1	-3	0	0	0	10000	0	4	0	0	11662...	4997	510
14	1	-3	0	0	0	10000	0	4	0	0	11662...	4520	536
15	1	-3	0	0	0	10000	0	4	1	0	11662...	-14898	1986
16	1	-3	0	0	0	10000	0	4	1	0	11662...	-14877	1983
17	1	-3	0	0	0	10000	0	4	1	0	11662...	-14865	1982
18	1	-3	0	0	0	10000	0	4	1	0	11662...	-14910	1988
19	1	-3	0	0	0	10000	0	4	1	0	11662...	-15174	2023
20	1	-3	0	0	0	10000	0	4	1	0	11662...	-14811	1974
21	1	-3	0	0	0	10000	0	4	1	0	11662...	-14934	1991
22	1	-3	0	0	0	10000	0	4	1	0	11662...	-15081	2010
23	1	-3	0	0	0	10000	0	4	1	0	11662...	-15174	2023

14. ábra – Egy verzió sorai

Ha adatok importálása előtt kiválasztunk egy modellnevet és verziót, azt a program úgy értelmezi, hogy oda akarjuk az adatokat elhelyezni, így automatikusan kitölti a modell név és verzió mezőket a párbeszédablakban (ezek persze változtathatók az importálás során).

3.8 Eredmények törlése

A  gomb segítségével törölhető az adatbázisból minden adat, amely az éppen kiválasztott batch-hez, verzióhoz, vagy modellhez tartozik. Amennyiben egy verziót jelölünk ki, akkor az alá tartozó összes batch adatai törölődnek; modell kiválasztása esetén pedig a hozzá tartozó összes verzió eredményei kerülnek eltávolításra.

4 Nézetek


4.1 Koncepció: nézet-tábla

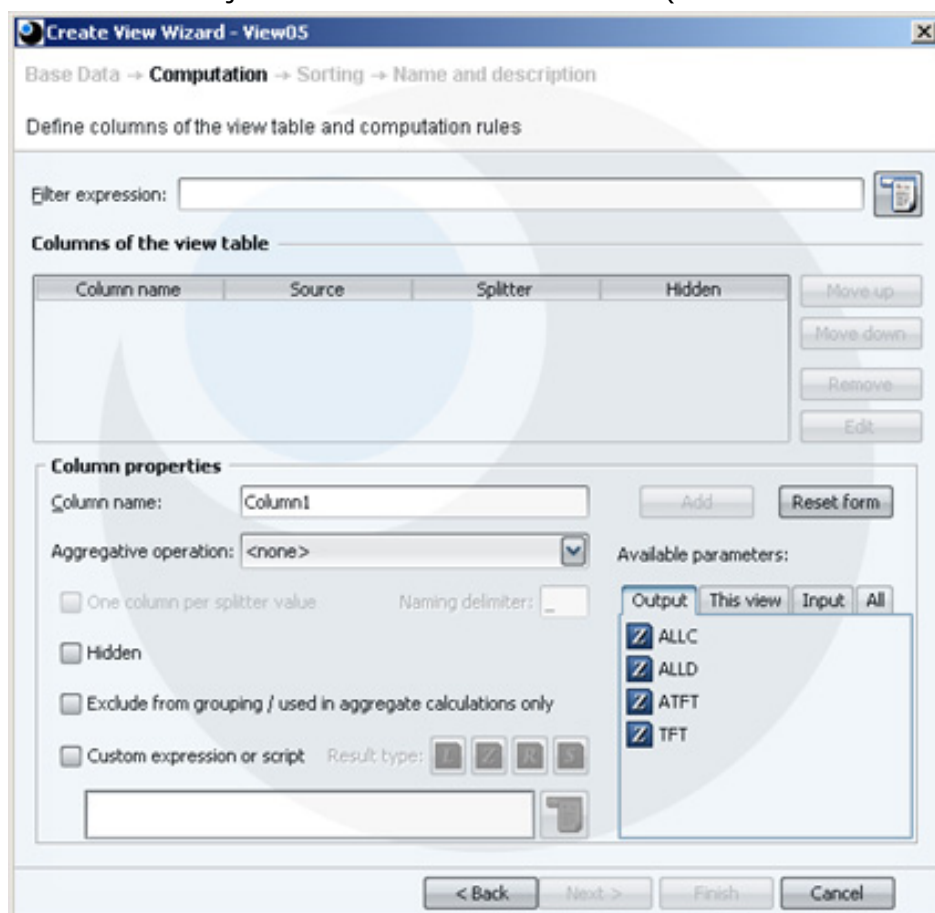
A nézet-táblák nyers szimulációs adatokból készített, letisztított adattáblák. Tetszőleges modellből és verzióból származó adatkezegekből (batch-ekből) készíthető a nézet-tábla, és ahhoz a kiinduló táblák tetszőleges oszlopa hozzáadható. Az eredeti adatokon számítások és szűrések hajthatóak végre (lásd 4.2.1 1. lépés – számítások!).

Grafikonok csak nézet-táblákból hozhatók létre. A MEME lehetőséget biztosít a nézet-táblák CSV formátumba történő exportálására is (lásd 3.6 Fájlexportálás!).

4.2 Nézetek létrehozása

Nézet-tábla létrehozásához először is ki kell választani egy batch-et a *Results browser*-ben. Az új nézet-tábla a kiválasztott futási eredmények adataiból fog létrejönni.

Ezután válassza a *Views/Create view...* menüpontot, vagy nyomja meg a  gombot a *Results* panelen. Ez elindítja a nézet létrehozása varázslót (*Create View Wizard*):







15. ábra - Számítások

Megjegyzendő, hogy már létező nézet-táblából is hozható létre nézet-tábla, ehhez a nézet kiválasztása után meg kell nyomni a *Create Views* gombot.

A varázsló használatakor a billentyűzet *Del* gombjának lenyomása ugyanazt eredményezi, mint a *Remove* gomb megnyomása a programban.

4.2.1 1. lépés – számítások

Oszlop hozzáadásához ki kell választani egy paramétert a *Available parameters* listáról. A lista fel van osztva a modell *Input*, illetve *Output* paramétereire, és a már használatban lévő paraméterekre. Az *All* fület választva megtekinthető az összes paraméter.

Az oszlopnevek mellett álló , , ,  ikonok jelzik az oszlop adattípusát: egész, valós, szöveg, vagy logikai érték.

Az oszlop kijelölése után lehetőség van egyszerű aggregációs függvények (Min, Max, Average, Sum, Count) teljes oszlopon történő alkalmazására. Az adott paraméterre beállítható, hogy maradjon ki a csoportosításból, így az oszlop értéke nem változik meg a táblával végzett műveletek során. Egyedi kifejezések (expression) vagy scriptek (Beanshell) futtathatók a változókon a program által nyújtott műveleteknél bonyolultabb számítások elvégzésére. Mivel a tick-ek és futások a szimulációs adatok sarokkövei, a MEME rendelkezik ezeket visszaadó, beépített scriptekkel; *\$Tick\$* és *\$Run\$*. Ha a szűrés alapjául szolgáló adat nézet-táblából származik (lásd 4.2.5 0. lépés – kiindulási adat!), akkor a *\$Tick\$* a sorok számaival tér vissza.

A *Hidden* opció választásával a megfelelő oszlop nem lesz látható a nézet-táblában. Ha egy oszlopot splitter-nek választunk, akkor a *Hidden* opció is automatikusan kiválasztásra kerül (ez később változtatható).

Alapértelmezett viselkedésként az oszlopnevek a megfelelő paraméterek nevei lesznek, de ez változtatható: nevet a *Column name* mező kitöltésével lehet megadni.

Kérjük, figyeljen a következőkre:

- Az oszlop neve egyedi kell, legyen (kis- és nagybetű különbözik), és nem lehet hosszabb 64 karakternél.

Az *Add* gomb megnyomásával az oszlopot hozzáadjuk a táblához. Ez után az oszlop használható *Splitter*-ként: ekkor a nézet többi oszlopa ennek az oszlopnak az értékei szerint kerül csoportosításra. Splitter oszlop definiálása után a többi oszlopra megadható a *One column per splitter value* opció.

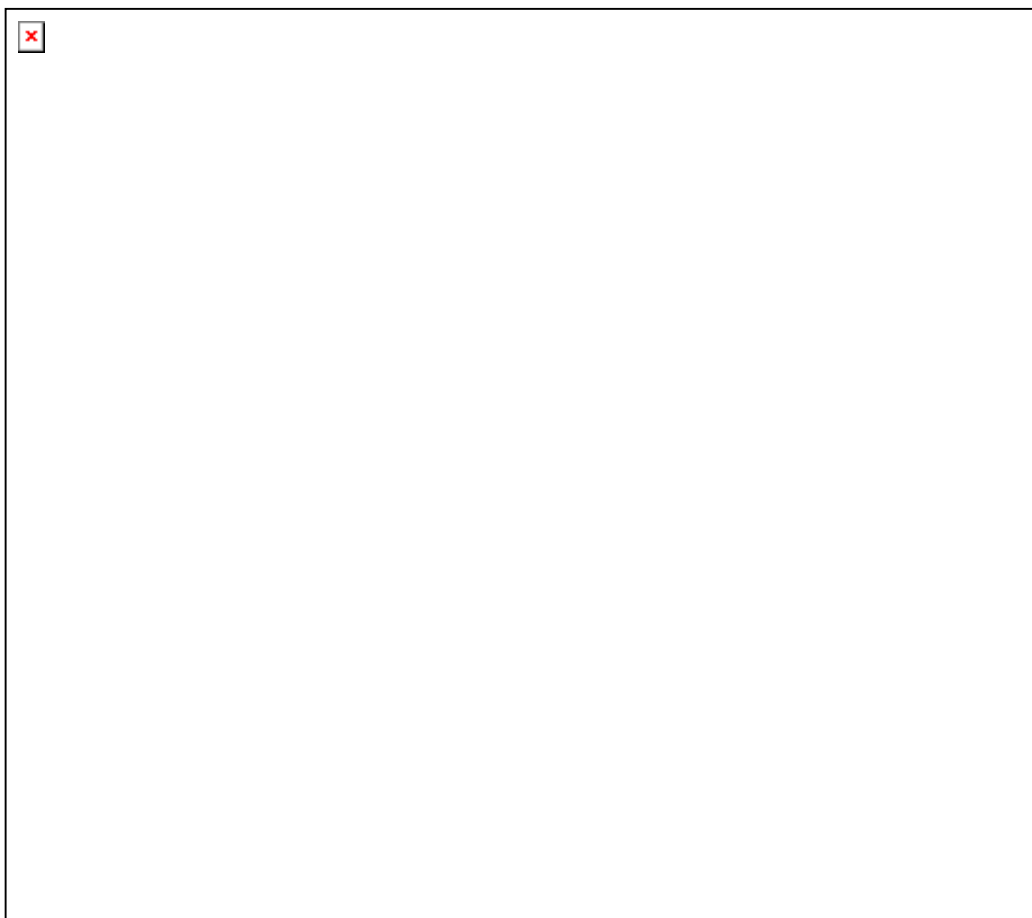
Az *Edit* gomb megnyomásával, illetve a kiválasztott (előzőleg hozzáadott) oszlop nevére duplán kattintva az oszlop beállításai szerkeszthetők: az oszlop átnevezhető, műveletek végezhetőek rajta, csoportosíthatóvá tehető, és splitternek választható. Szerkesztés után a változtatások a *Modify* gomb megnyomásával menthetők, vagy a *Cancel* gombbal elvethetők. Az oszlopok sorrendjének módosításához, illetve oszlop eltávolításához a *Move up*, *Move down*, és a *Remove* gombok használhatók.

A változók értékei szűrhetők a *Filter expression* mezőbe írt reguláris Java kifejezések segítségével.

Megjegyzendő, hogy a listában kiválasztott paramétert, illetve a módosítás alatt lévő oszlop sorait a program kiemeli (az alapértelmezett színsémával és témával narancssárga színnel).

Az oszlopok sorrendjének változtatásához egyszerre több oszlop is kijelölhető, ilyenkor az összes kijelölt sor feljebb, vagy lejjebb kerül. Megjegyzendő, hogy több kijelölt sor esetén az *Edit* gomb lenyomása után a kijelölés első oszlopa lesz szerkeszthető. Az oszlopok szerkesztése során a mozgatási és *Remove* gombok nem használhatóak.

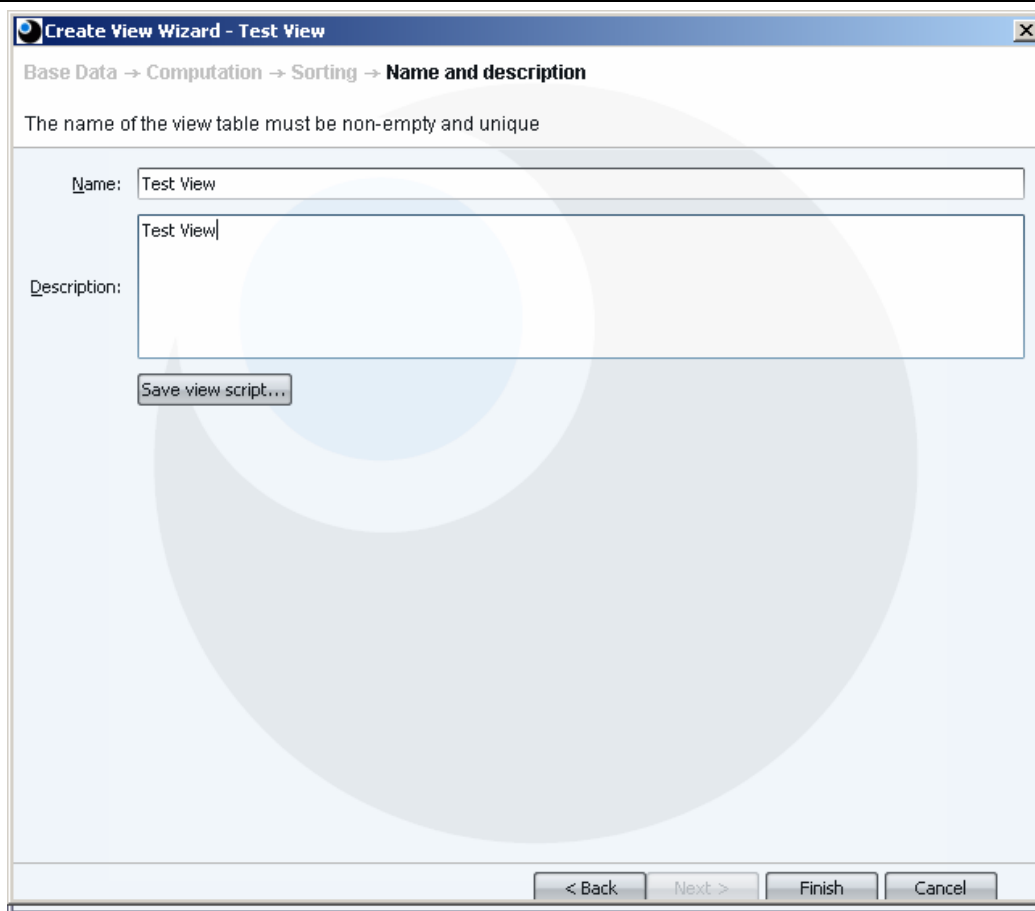
4.2.2 2. lépés - rendezés



16. ábra - Rendezés

A Rendezés ablakban beállítható a nézet-tábla rendezése a többi oszlopot csoportosító oszlopok sorrendjének megadásával. A csoportosító oszlopoknál növekvő, illetve csökkenő sorrendet lehet definiálni. A nézet-tábla rendezésének változtatása az oszlop kiválasztása után a *Move up* és *Move down* gombok segítségével történik.

4.2.3 3. lépés – név és leírás



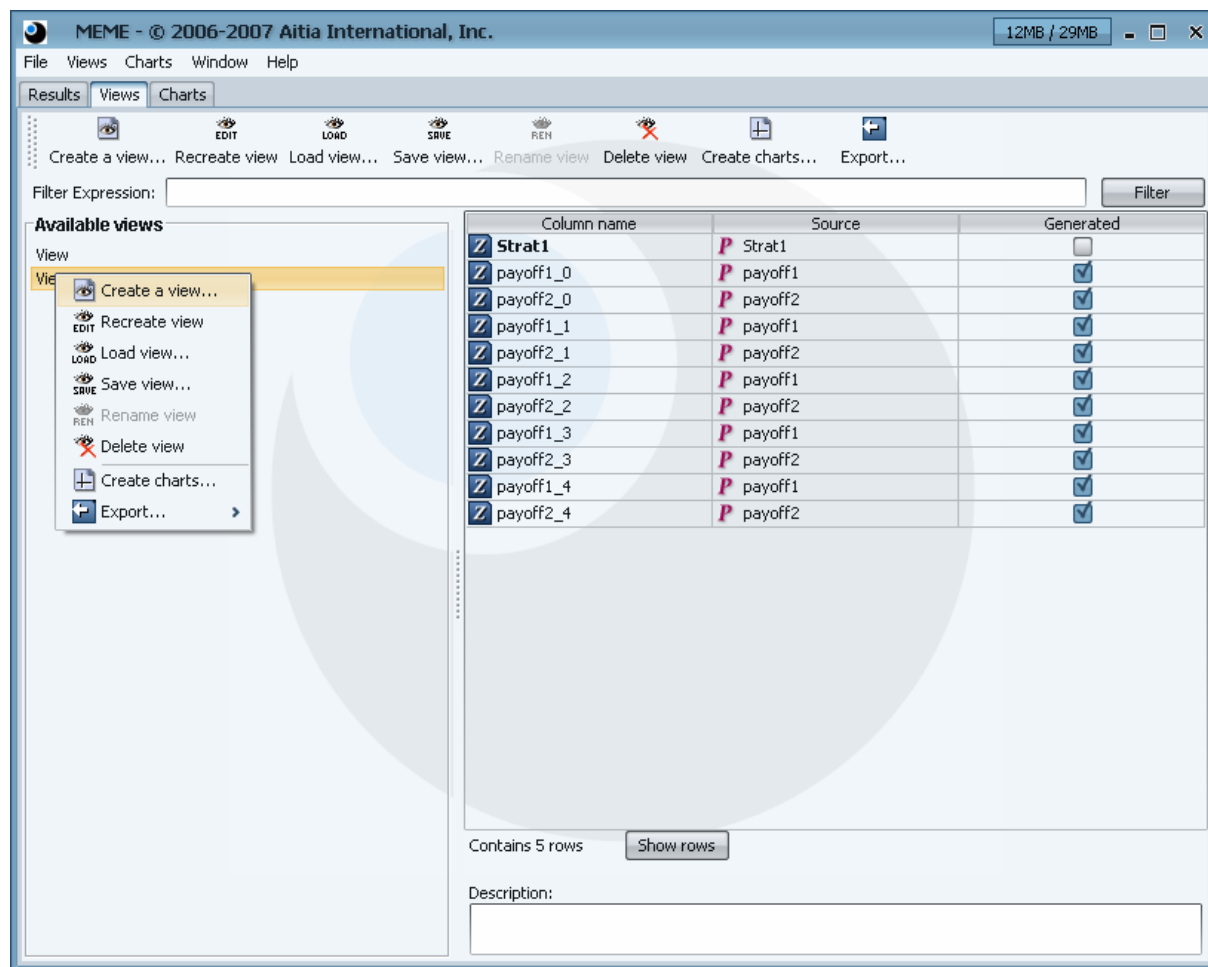
17. ábra - Név és leírás

Adjon meg egy - maximum 64 karakter hosszúságú - nevet a *Name* mezőben. Ha már létezik ilyen néven nézet-tábla, akkor azt a *Finish* gomb lenyomásakor a program felülírja (a program figyelmeztet az azonos nevű nézet létezésére). A *Description* mezőben további leírást adhat méretkorlátok nélkül. Nyomja meg a *Next* gombot a továbblépéshez.

A *Save view script* megnyomásával a nézet-táblát leíró script fájlba menthető. További információért lásd a 6 Scriptelés részt!

4.2.4 Az eredmény

A *Finish* gomb megnyomása után a *Wizard* bezárul, és új nézet-tábla jelenik meg a *Views* fülön. A MEME szintaktikailag ellenőrzi a nézet elkészítéséhez használt scriptek helyességét: ha hibát talál, akkor ezt hibaüzenettel jelzi, és a *Wizard* nyitva marad, lehetőséget adva a hiba javítására. Ilyenkor a *Cancel* gomb megnyomására a MEME egy üres nézet-táblát készít, ami azonban tartalmazza az összes beállítást. Ez lehetővé teszi a felhasználó számára, hogy bezárhassa a *Wizard*-ot, és a probléma elhárítása után újraindítva azt befejezhesse a nézet elkészítését.



18. ábra - Nézetek

Az *Available views* lista tartalma szűrhető a *Filter Expression* mezőbe beírt reguláris kifejezés segítségével, illetve a *Filter* gomb (vagy az *ENTER* billentyű) lenyomásával.

A fenti ábrán az összes nem-rejtett oszlop látható, és a splittelés hatására a *payoff1* és *payoff2* változókból öt-öt példány szerepel. Megjegyzendő, hogy splittelés esetén legfeljebb 10,000 oszlopos lehet a tábla (beleértve a rejtett (*hidden*) és a csoportosításból kihagyott oszlopokat is).

A splitter oszlopok (melyek a többi oszlopot csoportosítják) nevei félkövér betűtípussal vannak megjelenítve, míg a csoportosított oszlopok nevei normál betűtípussal. A *Generated* oszlopban szereplő pipa azt jelzi, hogy az adott oszlop splittelés során került létrehozásra.

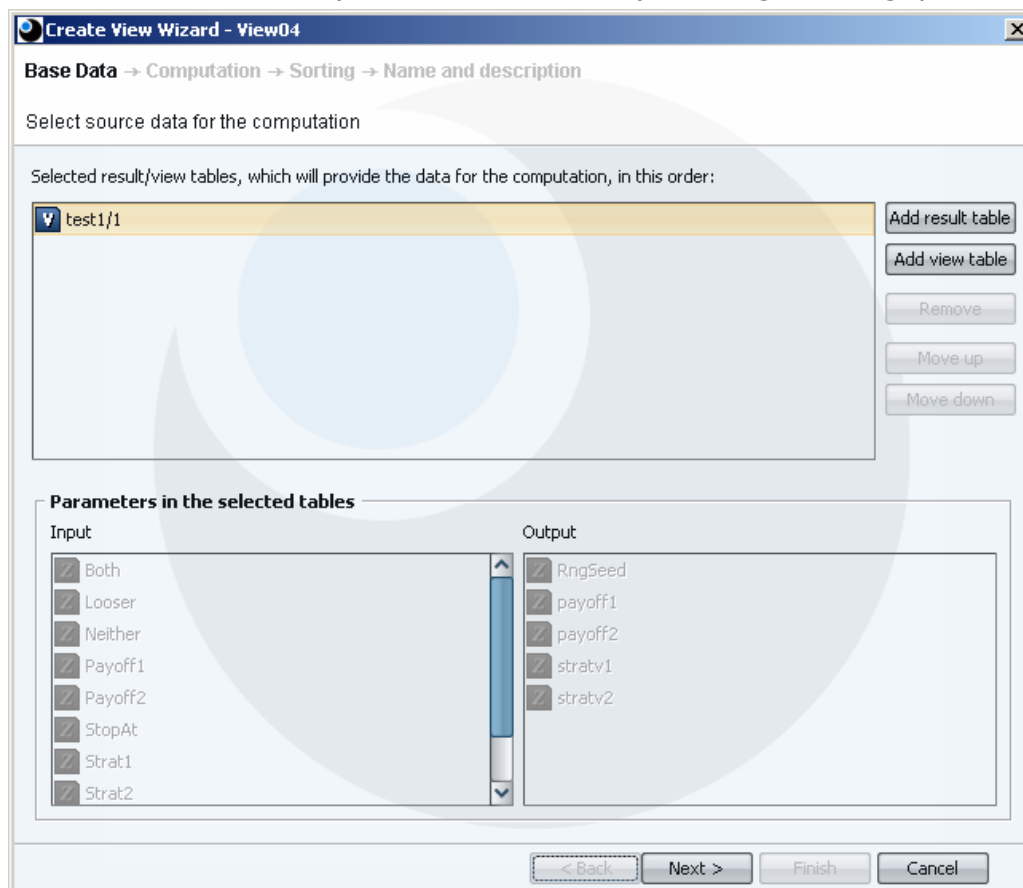
Az aktuális nézet-tábla adatai megtekinthetők a *Show rows* gomb megnyomásával, míg a kapott megjelenítésből a *Show columns* gombbal lehet visszaváltani. A nézet-táblából grafikonok készíthetők (lásd 5 Grafikonok rész!), vagy a tábla adatai exportálhatók más programokkal történő analízáláshoz, megjelenítéshez (3.6 Fájlexportálás).

A törlés, mentés, és exportálás műveletek egyszerre több nézet-táblán is végrehajthatók: több nézet kijelöléséhez a *CTRL* billentyű lenyomása mellett a megfelelő nézetek neveire kell kattintani. Több kijelölt nézet esetén az utoljára kijelölt nézet-tábla adatai jelennek meg a jobb oldalon: ilyenkor a *Recreate* gomb megnyomásával az utoljára kijelölt nézet kerül szerkesztésre.

Az oszlopok átnevezhetők a *Column name* mezőben a nevükre duplán kattintva. Ennek a szolgáltatásnak a segítségével apró simítások végezhet a nézet-táblán anélkül, hogy a táblát újra kéne generálni. Megjegyzendő, hogy ha egy nézet oszlopai ilyen módon kerültek átnevezésre, akkor a nézet újra elkészítése (*recreate*) problémákhoz vezethet, emiatt nem javasolt.

4.2.5 0. lépés – kiindulási adatok

A *Create View Wizard* a második oldalának megjelenítésével indul, mivel a *Wizard* indításakor már van kiválasztott eredmény tábla, így a kiindulási adatok már adottak. Haladó felhasználók számára lehetőség van további eredmény táblák, esetleg már elkészített nézet-táblák hozzáadására a kiindulási adatokhoz: az ehhez szükséges oldal elérhető a számítások oldalról (a *Wizard* kezdő oldala) a *Back* gomb megnyomásával.



19. ábra – Kiindulási adatok

Eredmény, vagy nézet-tábla hozzáadásához a megfelelő gomb megnyomása után a felugró ablakban ki kell választani a kívánt táblát, majd a *Select* gomb megnyomásával lehet hozzáadni a kiindulási adatokhoz: a felugró ablakban lehetőség van egyszerre több tábla kiválasztására is a *CTRL* gomb folyamatos nyomva tartása mellett.

4.3 Nézetek törlése

A nézetek a *Views* → *Delete* menüparanccsal, vagy a *DEL* billentyűvel törölhetőek ki. Ennek végrehajtása során a nézet-tábla minden adata törlődik: a művelet nem vonható vissza. A törlés funkció csak akkor érhető el, ha *Views* panelen ki van választva egy nézet.

4.4 Nézetek szerkesztése (Recreate)

A *Views* → *Recreate* menüparanccsal a már korábban létrehozott nézet-táblák adattartalma bővíthető, illetve szűkíthető. Új adatok adhatóak hozzájuk, avagy elvehetőek belőlük az adattábla létrehozását leíró szabályok változtatásával. A művelet akkor indítható, ha egy nézet ki van választva. A *Create View Wizard*-ot használja, amelyet korábban tárgyaltunk a 4.2 Nézetek létrehozása részben. A MEME tárolja, hogy az egyes nézet-táblák hogyan lettek létrehozva, beleértve azt is, hogy mely batch-eket használta, illetve milyen műveleteket hajtott végre az oszlopok létrehozásához. Ezeket a beállításokat lehet megváltoztatni a varázsló segítségével. Végül a nézet-tábla újra létrejön a jelenleg érvényes adatokkal és beállításokkal, felülírva a korábbi, ha a tábla

nevét nem változtatjuk meg. A tábla nevének megváltoztatása hasznos lehet például a szimuláció eredményeinek összehasonlításakor.


A *Recreate* eljárás meghívható a nézet-táblák listájában az adott nézetre duplán kattintva is.

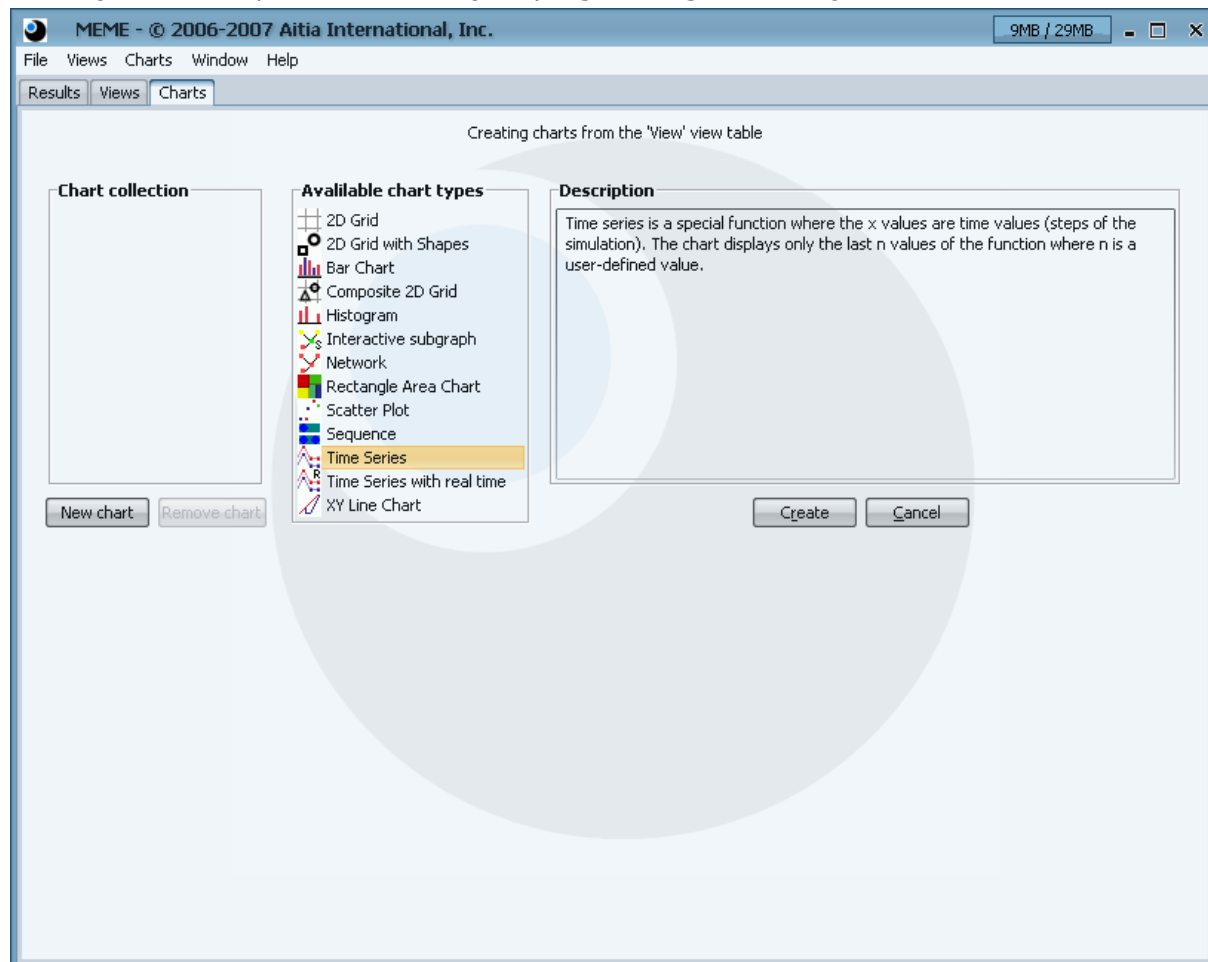
Megjegyzendő, hogy a nézetkészítés és -szerkesztés eljárások nem teljesen azonosan kezelik a rejtett oszlopokat. Ha egy oszlop szerkesztés közben rejtetté (*hidden*) válik, nem marad ki automatikusan az elkészülő nézet-táblából, mivel az elsődleges szempont az, hogy a már meglévő rendezések működjenek; az oszlop kihagyásához ki kell törölni azt.

5 Grafikonok

A MEME-ben grafikonokat csak nézet-táblákból lehet létrehozni, így grafikonok létrehozása előtt nézet-táblát kell készíteni a szimulációs eredményekből.

5.1 Grafikonok létrehozása (Charts/Create chart...)

Amint van már legalább egy nézet-tábla a programban, létre lehet hozni vizualizációkat. Ehhez ki kell választani egy nézetet az *Available views* listából a *Views* panelen, majd lenyomni a  gombot vagy elindítani a *Charts/Create chart...* menüparancsot. Ez aktiválja a *Charts* panelt, és átváltja a programot grafikon-rajzoló módba:



20. ábra - Grafikon létrehozása

Egy nézet-táblából számos vizualizáció készíthető. A létrehozott grafikonok az ablak bal oldalán, a *Chart collection* listában kerülnek felsorolásra. A létrehozható grafikontípusok középen, az *Available chart types* listában láthatóak, míg az egyes típusok leírása a jobb oldalon jelenik meg. A grafikontípusok közül egyet kiválasztva és a *Create* gombot lenyomva létrejön egy grafikon, a neve megjelenik a *Chart collection*-ben, a beállítási felülete pedig a jobb oldalon.

A *New chart* gomb használatával térhetünk vissza az *Available chart types*-hoz, és adhatunk újabb grafikont a gyűjteménybe. A *Remove chart* gomb kitörli a kiválasztott grafikont. Egy adott grafikont kiválasztva a *Chart collection* listából, megjelennek tulajdonságai és beállításai, amelyek változtathatóak.

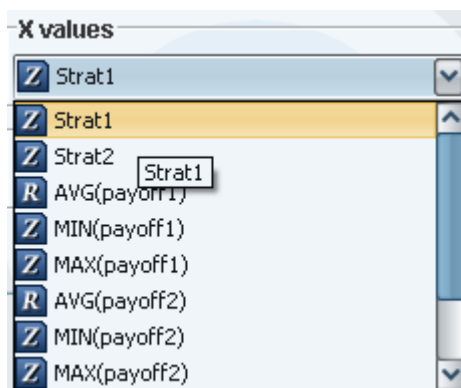
A *Cancel* gombra kattintva a program visszalép a grafikon rajzoló módból a *Views* panelre, és az el nem mentett grafikonok elvesznek.

A következőkben leírjuk a grafikon rajzoló üzemmód általános tulajdonságait. Ezek megértése fontos az egyes grafikontípusok részleteinek megismeréséhez, amiről bővebben az 5.9 Grafikontípusok részben lehet olvasni.

5.2 Koncepció: Adatforrások

Minden grafikon beállítások paneljén található egy (vagy több) legördülő lista, melyben a nézet-tábla oszlopai közül lehet választani. A kiválasztott oszlopból származnak majd a neki megfelelő változó adatai a grafikonon. A speciális # tétel a nézet-tábla sorszámait tartalmazza. Nullával kezdődik és minden sornál eggyel nő.

A grafikon típusától és az adott változótól függően lehetnek eltérések a kiválasztott nézet-tábla oszlop tényleges adattípusa és a grafikon által megkövetelt változó típusa között. Ezért, ha szükséges, az adatokon végrehajtásra kerül egy automatikus konverzió az alábbiak szerint:



Megkövetelt típus	Tényleges típus	Konverziós szabály
integer	valós	A valós számokat lefelé kerekíti.
szám ⁴	logikai	Az igaz 1-gyé, a hamis 0-vá konvertálódik.
Szám ⁴	szöveg	Lásd lent ⁵
szöveg	Szám ⁴	A számok szöveggel kerülnek kiírásra.
szöveg	logikai	Az igaz 1-gyé, a hamis 0-vá konvertálódik.

Az egyes grafikonfajták és adattípusok által „megkövetelt típus”-ról bővebben az 5.9 Grafikontípusok részben olvashat. Az adott adattábla „tényleges típusai” a Views panelen láthatóak (lásd a Z, R, S, L ikonok leírását a 4.2.1 részben!).

5.3 A Compose, Display, Save és Cancel gombok

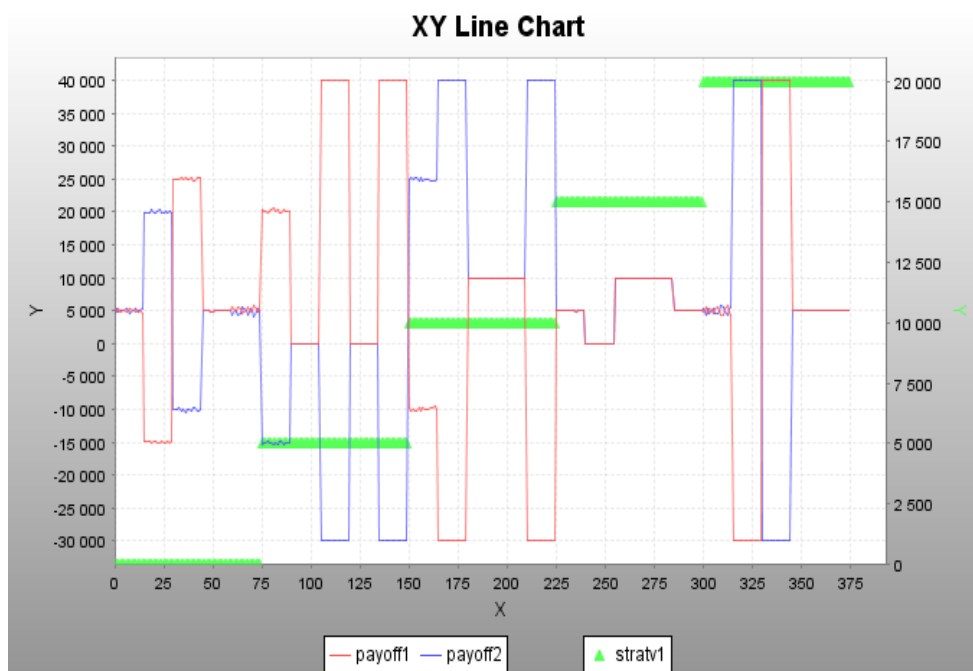
Ezek a gombok a grafikon beállítások ablak alján találhatóak. A *Display* gomb megjeleníti a grafikon egy új ablakban. Csak akkor lehet erre a gombra kattintani, ha minden szükséges információt megadtak, azaz a MEME-nek elegendő információja van egy grafikon megjelenítéséhez. A *Save* gomb elmenti a *Chart collection* összes grafikonjának beállításait egy fájlba. Megjegyzendő, hogy csak a grafikonok konfigurációját menti, nem az adatokat, amelyekből összeáll a grafikon.

A *Cancel* gomb azonnali hatállyal, mentés nélkül bezárja a grafikonrajzoló üzemmódot, bezárja az összes grafikon megjelenítő ablakot, és visszavált a Views panelre.

A *Compose* gomb lehetőséget ad arra, hogy több különböző típus kombinációjából hozzunk létre grafikonot. Például az alábbi grafikon egy XY Line Chart és egy Scatter Plot kombinációjaként jött létre. A *Compose* a következő grafikontípusokra elérhető: XY Line Chart, Time Series (mindkét típus!), Scatter Plot és Histogram.

⁴ A szám valós vagy integer értéket jelöl

⁵ Minden szöveghez egy egész szám rendelődik: a sorozat (ábécé szerinti) rendezése és a megegyező elemek eltávolítása után a program minden szöveghez a sorszámát rendeli. Ez a módszer lehetővé teszi szöveget tartalmazó oszlopok alkalmazását olyankor is, amikor számértékre van szükség.



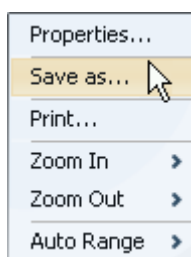
21. ábra – Egy kombinált grafikon

5.4 Nagyítás

A grafikon részleteinek megvizsgálásához kinagyíthat egy tetszőleges részt belőle. A bal egérgomb lenyomásával és balról jobbra lefelé mozgatásával jelölhető ki a nagyítani kívánt rész. Az eredeti nézethez való visszatéréshez az előző művelet ellenkezőjét kell végrehajtani, tehát a bal egérgomb lenyomása mellett az egeret balra felfelé mozgatni.

5.5 Ábrák mentése

Minden grafikon ablak rendelkezik egy menüvel, amely az ábrán történő jobb egérgomb kattintással hozható elő. Ez tartalmaz egy *Save as...* menüparancsot, mely segítségével elmenthető a grafikon PNG vagy EPS formátumban.



22. ábra – Helyi menü

A *Properties* parancs megjelenít egy párbeszédablakot, amelyben a grafikon alapbeállításait – cím, háttérszín, tengely címke, stb. – lehet megváltoztatni.

5.6 Tulajdonságok (Properties)

A *Properties...* ablak segítségével az ábra címét, és címkéit lehet szerkeszteni, beállítható a betűtípus, a keret, a háttérszín, illetve a tengelyek tulajdonságait is meg lehet változtatni. Az ablakot az előző szakaszban leírt módon lehet elérni.

5.7 A Details felület

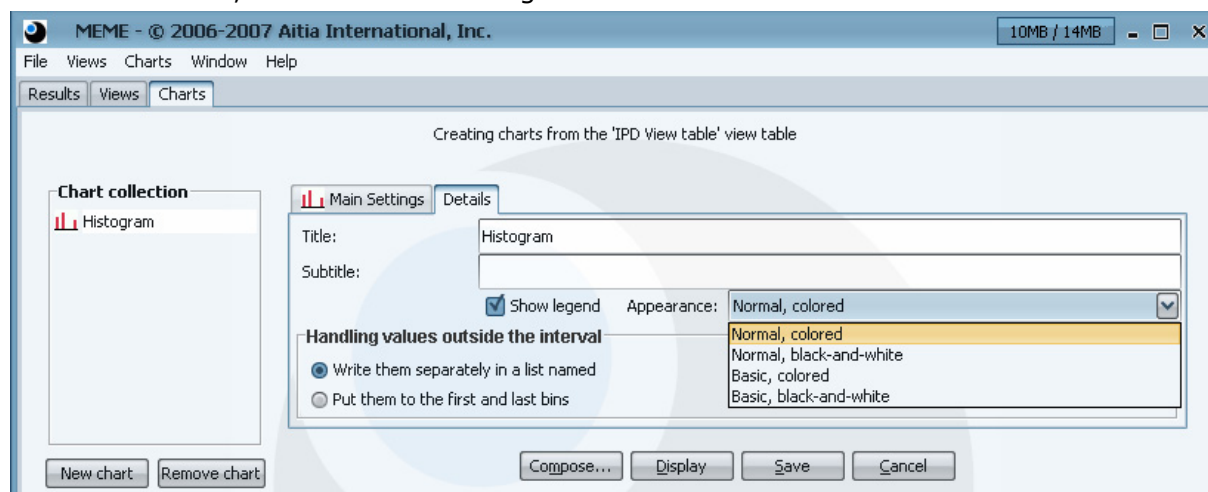
A grafikon beállításai két részre vannak osztva, a *Main Settings*-re (fő beállítások) és a *Details* (részletek) felületre (néha továbbiak is láthatóak). A fő beállítások rész azokat a minimum információkat tartalmazza, amelyek a grafikon létrehozásához

elengedhetetlenül szükségesek. A további beállításokat – mint a cím, tengely címke, színek, stb. – a *Details* és az egyéb felületeken lehet megtenni.

A grafikon megjelenése is beállítható a *Details* felületen, a választható beállítások:

- *Normal, colored*: színátmenetes színes grafikon
- *Normal, black-and-white*: színátmenetes fekete-fehér grafikon
- *Basic, colored*: színes, de egyszerű ábra
- *Basic, black-and-white*: egyszerű, fekete-fehér cikkekhez szánt ábra

Megjegyzés: néhány grafikontípusnál (pl. 2D Grid) a megjelenési beállítások külön felületen vannak, illetve a beállítások grafikononként eltérhetnek.



23. ábra - Details felület

5.8 Grafikon megnyitása

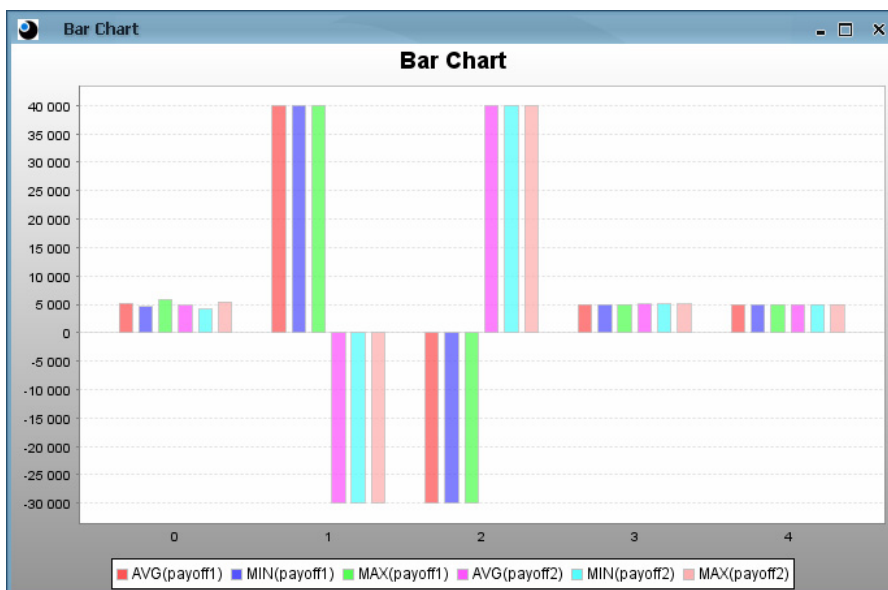
A *Charts* → *Open chart* menüpont lehetőséget biztosít egy korábban mentett grafikon gyűjtemény (chart collection) megnyitására, annak további változtatása, és a grafikonok megjelenítése céljából. Megjegyzendő, hogy a fentiekhez szükséges, hogy a nézet-tábla azonos néven és azonos oszlopokkal - amelyből a grafikonok létrehozásra kerültek - még mindig megtalálható legyen az adatbázisban, mert az adatokat nem menti a program a grafikon beállítások tárolásakor. Amennyiben újragyártásra került a nézet-tábla úgy, hogy az több vagy kevesebb adatot, vagy több oszlopot tartalmaz, akkor a megnyitott grafikon gyűjtemény megjelenítésre kerül az új adatokkal. Ugyanakkor, ha a nézet-tábla vagy a grafikonok által használt oszlopok törlésre kerültek, a megnyitás hibaüzenetet fog eredményezni.

5.9 Grafikontípusok

5.9.1 Oszlopdiaagram (Bar Chart)

Ez egy klasszikus grafikontípus, amely lehetővé teszi valós értékek sorozatainak közös kategóriahalmazon való összehasonlítását. Így szükség van egy adatforrásra a kategóriákhoz, és egyre vagy többre az összehasonlítandó értékekhez (ezeket adatsoroknak – *Data Rows* – nevezzük). A kategória adatforrás megkövetelt típusa szöveges, értékei a vízszintes tengelyen kerülnek listázásra. Az adatsorok értékei valós számok kelljenek, hogy legyenek, ezeket reprezentálják az oszlopok a grafikonon. Minden kategóriában az oszlopok egy csoportja látható, és minden csoportban egy-egy (eltérő színű) oszlop jelöl egy-egy adatsort. Oszlopdiaagramot más módon is meg lehet jeleníteni: készíthetünk halmozott (minden kategóriához egy osztott oszlopa van), és 100%-ig halmozott oszlopdiaagramot is.

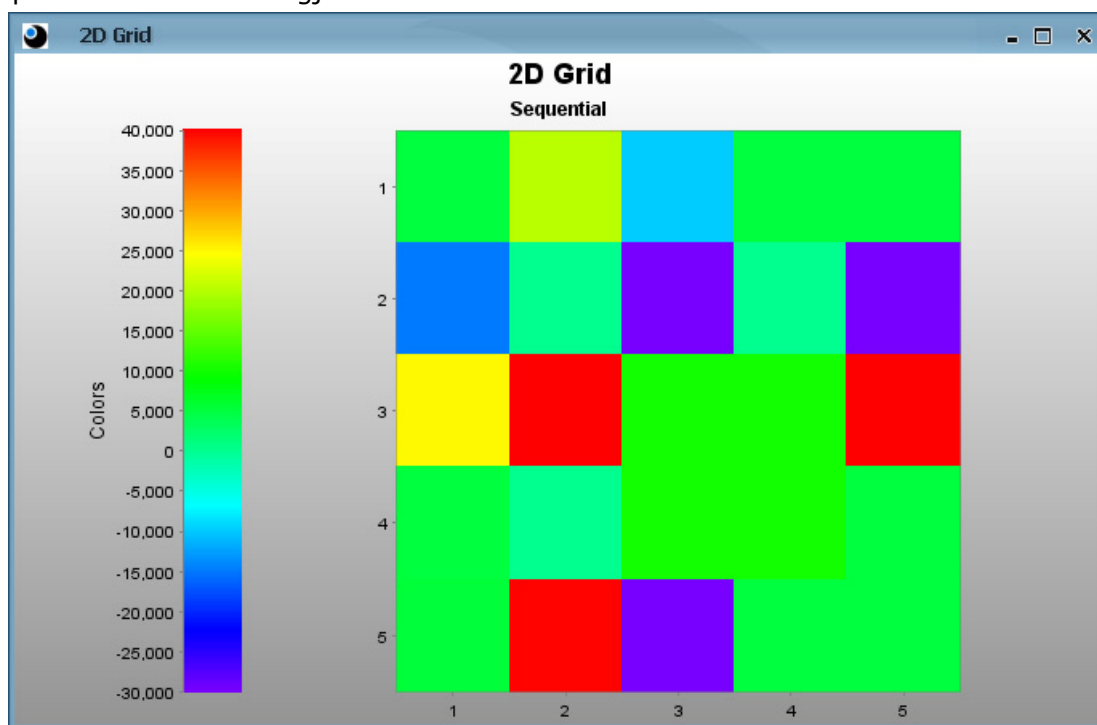
Ajánlott, hogy a kategóriák eltérőek legyenek, mivel az azonos szöveges értékek közül csak az utolsó, illetve az ahhoz tartozó oszlopok kerülnek kirajzolásra.



24. ábra – Egy oszlopdiagram

5.9.2 2D rács (2D Grid)

A 2D rács egy olyan diagram, amely színek mátrixából áll össze. Igen hasznos változók komplex halmazainak megjelenítésekor.



25. ábra – Egy 2D rács szekvenciális kitöltéssel

A színezés lehet közvetlen elérésű, vagy szekvenciális.

- Közvetlen elérésű színezés esetén három adatforrásra van szükség. Ezek értékeiből $(x, y, szín)$ hármasokat képzünk a mátrix színezésének meghatározásához. Az x és y értékek megkövetelt típusa integer (egész szám), míg a szín értékek valós számok kelljenek, hogy legyenek. A mátrix méretének meghatározása ebben az esetben opcionális. A szélesség (*Width*) és a magasság (*Height*), ha megadásra kerülnek, arra hivatottak, hogy mellőzzék az általuk megadott intervallumon kívül eső (x, y) párokat.

- Ha a mátrix szekvenciális csak egy adatforrásra (valós szám) van szükség. Ennek értékei egyenként adják meg a cellák színét soronként (*left-to-right* sorrend), vagy oszloponként (*top-to-bottom* sorrend). Ebben az esetben a mátrix méretét mindenképpen meg kell adni, vagy mindkét, de legalábbis az egyik érték meghatározásával, mely esetben a másikat automatikusan számítja a program a nézet-tábla sorainak számából.

A *Switch to random filling* és *Switch to sequential filling* gombok segítségével lehet választani a kétfajta grafikon közül.

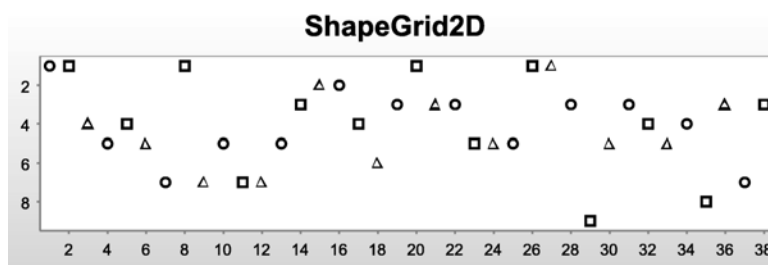
Alapbeállításként egy alkalmazkodó szivárvány színtérkép kerül alkalmazásra a valós értékek színezéséhez. Ez kék színt rendel a legalacsonyabb, piros színt a legmagasabb értékhez, míg a köztes értékeket egy szivárvány-skála alapján színezi ki, ahogy a fenti képen is látható. Ez a beállítás a grafikon beállítások *Colors* fülén változtatható meg. Az alternatívák a következők:

- **Rainbow colormap:** (szivárvány színtérkép): Szükséges egy minimum és egy maximum érték megadása, és a megfelelő színek meghatározása. A köztes értékek színezése lineárisan történik.
- **Heat colormap:** (hőterkép): A *Rainbow colormap*-hez hasonló színtérkép, ami fehértől sötétvörösre színezi.
- **Real colormap:** (valós színtérkép): Hasonló a *Heat colormap*-hez.
- **Pastel colormap:** (pasztell színtérkép): A *Rainbow colormap*-hez hasonló, pasztell színeket használó színtérkép.
- **Random colormap:** (véletlen-színtérkép): A *Rainbow colormap*-hez hasonló, de csak 36 színt használó színtérkép.
- **Simple colormap:** (egyszerű színtérkép): Egy minimum és egy maximum érték és nekik megfelelő színek megadása mellett a köztes értékek színezése lineárisan történik.
- **Table colormap:** (táblázatos színtérkép): Minden értékhez egy adott szín adható meg, a nem definiált értékekhez pedig egy alapszín párosul. A létrehozott színtérkép elmenthető egy fájlba, majd később újra betölthető. A fájl formátuma nagyon egyszerű, szükség esetén akár külső eszköz segítségével is létrehozható. Itt egy példa:

```
#
#Wed Dec 20 03:48:55 GMT+01:00 2006
67.0=-16776961
6.0=-52480
45.0=-6750055
87.0=-3342490
DEFAULT_COLOR=-1
```

5.9.3 Alakzat rács (Shape Grid)

Az alakzat rács nagyban hasonlít a 2D rácshoz, azonban színek helyett ennél rácsnál alakzatok párosulnak a valós számokhoz, úgy mint: \circ , \square , \triangle . Az adatforrások megkövetelt típusai nem különböznek az 2D rácsnál leírtaktól. Az értékekhez az alakzatok adott sorrendben kerülnek hozzárendelésre alapbeállításban, az ismétlődés megengedésével. A grafikonok beállításainál a *Rederer* fül segítségével megváltoztatható a hozzárendelés, érték-formaszín hármassok vagy érték-ikon párosok kézi beállításával; vagy betölthetők az *IFigureRenderer* interfészt megvalósító osztályok is.



5.9.4 Vegyes rács (Composite Grid)

A vegyes rács a 2D és az Alakzat egyszerű kombinációja. Ez azt jelenti, hogy négy adatforrásra van szükség a közvetlen elérésű rács esetében – $(x, y, szín, alakzat)$ négyesek formálásához – és kettőre a szekvenciális mátrix esetében – egy a cellák színekhez és egy másik az alakzatokhoz.

5.9.5 Hisztogram

A hisztogram a gyakoriság grafikai megjelenítése. Ez a grafikon egy numerikus adatforrás sűrűségét ábrázolja egy fix intervallumon. Az intervallumot n darab egyenlő részintervallumra („rekeszek”) osztja, és az értékek számát, amelyek egy-egy részintervallumba esnek megfelelő magasságú oszlopok segítségével jeleníti meg a grafikonon. Csak egyetlen, valós számokból álló adatforrást igényel ez a megjelenítés, de az intervallum alsó és felső korlátait, illetve az n értékét meg kell adni. Gyakoriság helyett lehetőség van az egy intervallumba eső értékek összegét, vagy átlagát is megjeleníteni.

A *Details* fülön be lehet állítani, hogy hogyan kezelje a program a megadott intervallumon kívül eső értékeket. Alapbeállításként ezek kizárólag a grafikon alján kerülnek listázásra. Beállítható, hogy ezekkel egyáltalán ne jelenítse meg, vagy az első, illetve utolsó rekeszben ábrázolja őket.

5.9.6 Hálózat (Network)

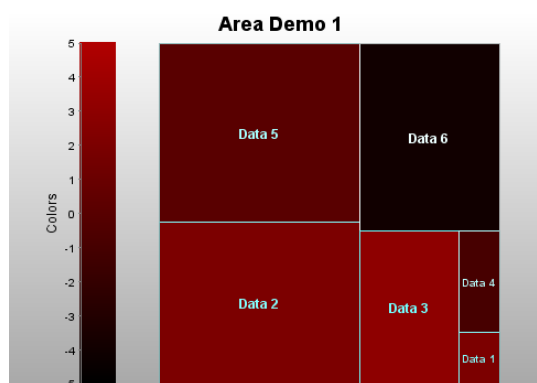
A hálózat (vagy gráf) elemek (irányított, vagy irányítatlan) élekkel összekötött halmaza. Az elemek neve csúcs vagy csomópont. Formálisan, a hálózat csúcsok és a csúcsok közti bináris relációk egy halmaza. A hálózat két számhalmazt vár (az élek kezdő és végcsúcsait) és egy szövegeket tartalmazó halmazt. A MEME hat különböző gráf rajzoló eljárást valósít meg (*kör*, *Fruchterman-Reingold*, *Spring*, *Kamada-Kawai*, *Kamada-Kawai 3D* and *ISOM (Meyer)*), illetve a gráf lehet irányított vagy irányítatlan.

Egy *Interactive subgraph* (interaktív részgráf) a gráf egy részét mutatja meg: a felhasználó megadhat egy csúcsot és egy mélységi korlátot, ekkor a részgráf a megadott elemet és a korlátnál nem messzebbi szomszédjait mutatja. A megadott csúcs dinamikusan változtatható, így lehetővé válik komplex gráfok interaktív bejárása.

5.9.7 Területalapú diagram (Rectangle Area Chart)

Ez a diagram hasonló a jól ismert kördiagramhoz, de négyzet az alapja kör helyett. Numerikus (valós) adatforrást használ, ami alapján a négyzetet arányosan felosztja kisebb négyszögekre. Egy második (számokat tartalmazó) adatforrás a színeket határozza meg, illetve egy harmadik (szöveges) adatforrás segítségével szöveget helyezhetünk el a téglalapokon. A színek valós értékekhez rendelése a 2D Rácsnál bemutatott szintértékes technikával történik.

A területalapú diagram a piaci analízis területéről származik, ahol „Map of the market”-nek („A piac térképe”) is nevezik. Ez a grafikon két különböző tulajdonsággal rendelkező adatsor megjelenítésénél hasznos.



5.9.8 Pontdiagram (Scatter Plot)

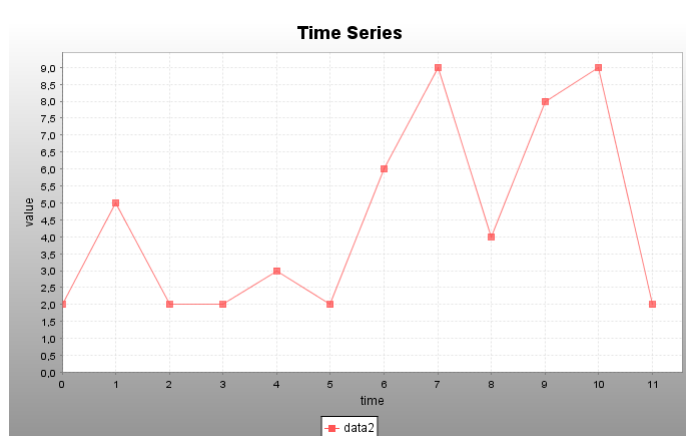
A pontdiagram egyszerűen (x, y) koordináta-párokat jelenít meg különálló pontokként egy koordináta rendszerben. A párok – az XY Line Chart-hoz hasonlóan – két numerikus (valós) adatforrásból, x és y értékekből áll.

5.9.9 Sorozat (sequence)

A sorozat grafikon halmazok rendezett sorozatának megjelenítésére szolgál (adatforrás csak számok rendezett halmaza lehet). Az elemek megjelenítése állítható: alapértelmezett, felhasználói beállítás, vagy egyedi kirajzolás használata mellett az elemek színe és/vagy megjelenése is beállítható.

5.9.10 Idősor (Time Series)

Ez a grafikontípus az utolsó n elemét jeleníti meg a sornak egy klasszikus vonaldiagramon. Az n értéke alapbeállításaként 10, de ez a *Main Settings* felületen állítható. A grafikon számértékeket vár (valós) adatforrásként – ezekből kerülnek ki az y értékek -, míg az x értékek automatikusan generálódnak az adatok sorozatban elfoglalt helyéből. Egynél több adatforrás is megjeleníthető a grafikonon, mely esetben különböző színekkel kerülnek az idősorok megjelenítésre.



A *Time Series with real time* (valós idejű idősor) egy olyan speciális függvény, ahol az x értékek automatikusan generálódnak a rendszeridőből.

5.9.11 Vonaldiagram (XY Line Chart)

Ez a klasszikus vonaldiagram. Két szám (valós) adatforrást igényel: egyet az X és egyet az Y értékek számára. Az eredményeként létrejövő $x \mapsto y$ leképezés egy vonal segítségével kerül kirajzolásra (ahol $x \in X$ értékek, $y \in Y$ értékek a nézet-tábla megegyező sorából) az idősorhoz hasonlóan, azonban az adatpontok jelölése nélkül. Az x értékeknek nem kell növekvő sorrendben lenniük, amennyiben szükséges, az (x, y) adat párok rendezésre kerülnek. Ugyancsak lehetséges, hogy egy grafikonon egynél több vonaldiagramot ábrázoljunk: az *Add* gomb használatával több X és Y adatforrás pár is definiálható.

6 Scriptelés

Megjegyzendő, hogy a scriptelés tapasztalt felhasználók számára ajánlott. Scriptek írásához szükség van programozói és alapvető xml ismeretekre.

A MEME két típusú scriptelést kínál:

- Mivel a nézet-táblákat és grafikonokat leíró adatok xml formátumban kerülnek tárolásra, ezért lehetséges a táblák xml fájlokba történő mentése, illetve xml fájlokból kiolvasása. Így több tábla módosítása egyszerűbben végrehajtható.
- Beanshell scriptek használhatók nézet-tábla készítésénél, bonyolult oszlop-számítások megadására.

6.1 Nézet scriptjének mentése

Egy, vagy több nézet scriptjének mentése a menteni kívánt nézet(ek) kiválasztása után a *Save view* gomb megnyomásával történik. A nézet scriptje a *Create View Wizard*-ban is elmenthető a *Save script* gomb megnyomásával (lásd 4.2.3 3. lépés – név és leírás!). Mentéskor a program a *<nézet-tábla neve>.xml* nevet ajánlja fel: ez csak egyetlen fájl (script) mentésekor változtatható, több fájl mentése esetén az alapértelmezett nevet kapják a fájlok.

Megjegyzendő, hogy a tábla scriptjének mentése nem egyenértékű a nézet-tábla exportálásával (lásd 3.6 Fájlexportálás!). Fájlexportálás során a nézet-tábla összes adata kiírásra kerül, mentéskor csak a nézetet leíró scriptet tároljuk el.

6.2 Nézet scriptjének betöltése

Egy, vagy több nézet-tábla betöltése a *Load view...* gomb megnyomásával történik a *Results* vagy a *View* felületen. Több nézet-tábla betöltése az *Open file* párbeszédablakban több xml file kiválasztásával lehetséges.

6.3 Nézet-táblák szerkesztése

A nézet-táblákat leíró scriptek mentésének és betöltésének fő motivációja a nézet-táblák módosítása. Nagyon sok, alig eltérő nézet-tábla egyenkénti létrehozása igen időrabló lehet a MEME-ben. Vagy, ha különböző adatforrású, de azonos beállítású nézeteink vannak, amelyeket nem kell változtatni, egyszerűen nem ésszerű újragyártani azokat. Ezekben az esetekben a nézeteket leíró scriptek átírása időt és energiát spórol meg.

A táblákat leíró xml fájlok felépítésének bemutatására szolgál a lentebb álló megjegyzésekkel ellátott példakód. A megjegyzések `<!--` jelsorozattal kezdődnek, `-->` jelsorozattal végződnek, és **kék** színnel kiemelték.

6.3.1 Példakód

```
<viewrule version="1.0.70221">
  <name> nézet neve </name>
  <description> leíró szöveg </description>
  <data>
    <table name="res" version="101-200" />      <!-- eredmény tábla -->
    <table name="view3" />                       <!-- nézet-tábla -->
    <!-- itt bármennyi eredmény vagy nézet-tábla megadható -->
  </data>
  <columns ordering="0 1" grouping="true">
    <!-- az 'ordering' attribútum megadása nem kötelező. A számok a lentebb
    megadott oszlopok sorszámjai, a 0 jelöli az első oszlopot. A mező értéke
    alapértelmezésben üres. A 'grouping' attribútum logikai (true/false)
    érték, alapértelmezésben hamis (false). -->
    <column datatype="12.64.0" grouping="false" hidden="false" agrfrn="AVG"
    splitter="false">
      <!-- a datatype="12.64.0" itt varchar(64)-et jelöl. A datatype értékét a
      ColumnType típus generálja, illetve értelmezi. -->
      <!-- a grouping="true" és az agrfrn"<>" megadása kölcsönösen kizárja
```

```

    egymást -->
<!-- ha a splitter="true", akkor grouping="true" -->
<name> colX </name> <!-- nézet-tábla oszlopának neve -->
<split></split>
  <!-- Csak bontott (split) oszlopoknál használatos, hidden="false" és
  grouping="false" értékeket eredményez -->
  <!-- Lehet több projection_* tag is, de utána nem szerepelhet *_script
  tag. Script tagból összesen egy lehet. -->
  <projection source="input"> anInputColumn </projection>
  <projection source="output"> anOutputColumn </projection>
  <projection source="view"> previousColumnName </projection>

  <scalar_script datatype=".."> beanshell kifejezés </scalar_script>
  <!-- 'datatype' csak akkor használatos, ha aggrfn!="" -->
  <group_script> kifejezés </group_script>
  <!-- group_script esetén a grouping="false" és aggrfn="" kötelező-->
</column>
</columns>
<condition>
  <scalar_script> col1 == 3 </scalar_script>
</condition>
</viewrule>

```

6.3.2 Megjegyzések

Adattípus kódolásánál (`datatype="12.64.0"`): Az első szám a típus kódja, a második és harmadik szám alkotja a paramétereket (például: a VARCHAR mérete; 0, ha nem használt paraméter). A típusok bővebb leírása a `java.sql.Types` csomagban található, az ehhez kapcsolódó kiegészítéseink a következők:

- **logikai érték** (boolean) ábrázolása `java.sql.Types.CHAR`-al történik
- **egészek** (integer) ábrázolása `java.sql.Types.INTEGER`-el történik
- **long típusú értékek** ábrázolása `java.sql.Types.BIGINT`-el történik
- **double típusú értékek** ábrázolása `java.sql.Types.DOUBLE`-el történik
- **szövegek** ábrázolása `java.sql.Types.VARCHAR`-al történik

Ha egy eredménytáblára hivatkozó script hivatkozása nézet-táblára változik, akkor az összes változó típusát `output` jellegűre kell állítani, hiszen nézet-táblákban nincsenek `input` jellegű változók. Az érintett oszlopoknál a `<projection source="input">` bejegyzést `<projection source="output">` bejegyzésre kell változtatni.

6.4 Beanshell scriptelés

Megjegyzendő, hogy a Beanshell scriptelés tapasztalt felhasználók számára ajánlott, mivel scriptek írásához szükség van programozói ismeretekre.

Részletesebb Beanshell kézikönyv a Beanshell honlapján található [5]. Ez az útmutató csak a *View Creation Wizard*-ban írandó scriptekhez hasznos részleteket tartalmazza.

A scriptek több sorosak is lehetnek. Visszatérési értékük mindig vagy az utolsó kiértékelt kifejezés vagy metódus hívás eredménye, vagy az az érték, amit a `return` utasítással adunk meg.

A MEME-ben két fajta script létezik: skalár, vagy tömb visszatérési értékű. Mivel a MEME minden paraméterből létrehoz egy globális változót, a scriptekben a paraméterekre a nevükkel lehet hivatkozni. Skalár visszatérési értékű script esetén a hivatkozott paraméter típusa egyetlen érték (`double`, `int`, `string`, stb.), míg a tömb visszatérési értékű scriptekben a hivatkozott paraméter tömb típusú (`double[]`, `int[]`, `string[]`, stb.). Például a `p+3` kifejezés csak skalár visszatérési értékű scriptben helyes, míg a `p.length` egy tömb visszatérési értékűben.

6.4.1 Beépített műveletek

Mivel a tick-ek és futások a szimulációs adatok sarokkövei, a MEME rendelkezik ezeket visszaadó, beépített scriptekkel; `$Tick$` és `Run`. A további beépített műveletek listája a következő:

- `get("name")`: visszatér a `name` nevű paraméterrel. A művelet által visszaadott érték és a `name` nevű paraméterből készített változó értéke eltérő lehet:
 - A `get("name")` mindig a paraméter eredeti értékével tér vissza, akkor is ha a `name` nevű globális változó értéke már megváltozott. (Kivéve tömb visszatérési értékű script esetén (mikor a változók tömbök), ha csak egyes elemek értéke változott, nem a teljes tömb.)
 - Ha a megadott név nem érvényes Java azonosító (például `Column0.1`), mivel ilyenkor nincs globális változó, emiatt a paraméter a nevével nem, csak a műveleten keresztül hivatkozható.

Ha nincs `name` nevű paraméter, vagy `null`, a script hibát ad.

- `geti("name")`, `geto("name")`, `getv("name")`: visszatér a `name` nevű input/output/this view jellegű paraméter értékével. Ezen függvények használata akkor szükséges, ha különböző jellegű, de azonos nevű paraméterek vannak. Ekkor a `name` globális változón keresztül, vagy a `get("name")` segítségével csak az első érhető el. A láthatósági sorrend megegyezik a *Wizard* listáinak sorrendjével: *Output*, *This view*, *Input*. Például a `name` nevű *Output* paraméter eltakarja a többi kategória azonos nevű paramétereit.
- `gett("name")`: a technikai paraméterek lekérdezése:
 - `batch`: az aktuális batch sorszáma, vagy `null`, ha az aktuális input sor nem kapcsolódik batch-hez (mert pl. view táblával kapcsolatos)
 - `run`: az aktuális run sorszáma, vagy `null`, ha az aktuális input sor nem kapcsolódik batch-hez
 - `tick`: eredménytábla esetében az aktuális tick, nézet-tábla esetén a sorszám (#)
 - `model`: az aktuális modell neve, vagy `null`, ha az aktuális input sor nem kapcsolódik modellhez
 - `version`: a modellen belül az aktuális verzió neve, vagy `null`, ha az aktuális input sor nem kapcsolódik modellhez
 - `starttime`: az aktuális run kezdési időpontja (long), vagy `null`, ha az aktuális input sor nem kapcsolódik eredménytáblához
 - `endtime`: az aktuális run befejezési időpontja (long), vagy `null`, ha az aktuális input sor nem kapcsolódik eredménytáblához
 - `view`: az aktuális view tábla neve, vagy `null`, ha az aktuális input sor nem kapcsolódik nézet-táblához
 - `isgroupfirst`: `true`, amikor csoportosítás van és az aktuális input sor a csoporton belül az első; minden más esetben `false`
 - `isgrouplast`: `true`, amikor csoportosítás van és az aktuális input sor a csoporton belül az utolsó; minden más esetben `false`
 - `isblockfirst`: `true`, amikor tagolás van és az aktuális input sor a tagon belül az első; minden más esetben `false`
 - `isblocklast`: `true`, amikor tagolás van és az aktuális input sor a tagon belül az utolsó; minden más esetben `false`
- `call("fn", args...)`: a `fn` nevű aggregációs művelet meghívása. Pl. `call("AVG", p1, p2)` eredménye a `p1` és `p2` paraméterek értékének átlaga. `args`-ban tetszőleges kifejezések felsorolhatóak, nemcsak paraméterváltozók. Ha 10-nél több van, akkor 1 db `new Object[] { ... }`-ben kell megadni őket. "fn" az aggregációs művelet neve, úgy ahogy az *Aggregative operation*: legördülő listában szerepel (a végén lévő „()”-al együtt vagy nélküle). Használható a teljes Java osztálynév is (az, amelyik az `ai.aitia.meme.pluginmanager.IVCPlugin` interfészt megvalósítja), pl. az `AVG()` műveletnél `"ai.aitia.meme.builtinvcplugins.Avg"`).

7 Ismert problémák

7.1 MEME Parameter Sweep Tool

A bevezetőben arról írtunk, hogy a MEME egy eszköz szimulációk batch módú futtatásának kezelésére. Jelenleg Repast modellek batch verzióját létrehozásához, illetve kísérleteket futtatásához a MEME Parameter Sweep Tool-t kell használni, melyet a MEME installere különálló alkalmazásként telepít gépére.

A PS Tool segítségével egyszerűen létrehozhatóak Repast modellek batch verziói, illetve futtathatóak azok akár helyi gépen, akár griden vagy klaszteren. Az eszköz használatáról bővebben a MEME Parameter Sweep Tool Kézikönyvben olvashat.

A szimulációk futtatása után a *File / Import / Repast result file* parancs segítségével importálhatók az eredmények MEME adatbázisába és kezdődhet meg feldolgozásuk.

A közeljövőben integrálni fogjuk ezeket a különálló alkalmazást a MEME-be, így a MEME automatikusan fogja az eredményeket összegyűjteni és adatbázisba rendezni.

8 Hibaelhárítás

8.1 Hibajelentések

A MEME folyamatoss fejlesztés alatt áll, így bármilyen visszajelzést és hibajelzést szívesen fogadunk, ugyanakkor új, javított verziók rendszeresen kerülnek kiadásra. Ha bármilyen probléma lépett fel a program használatával kapcsolatban, vagy ötlete, javaslata van azzal kapcsolatban, kérjük lépjen velünk kapcsolatba a megadott e-mail címen.

Amennyiben hibajelentést küld, kérjük csatolja a programkönyvtárból a MEME.log fájlt, és írja le a lépéseket, amelyeket hiba felmerülése előtt tett.

8.2 Memória használat

Az adatbázis műveletek számára fenntartott alapértelmezett maximális heap méret 256 MB. A teljes program memória igénye ennél több, akár 400 MB is lehet az alapértelmezett beállítások mellett. 100 000 sornál több adat esetén a heap mérete kevésnek bizonyulhat. Ha a számítógépben 1 GB, vagy több RAM van, akkor a heap mérete nyugodtan módosítható. A MEME ikonra jobb egérgombbal kattintva, a felugró menü *Properties* pontját választva megváltoztatható a *Target* mező értéke `"javaw.exe -version:1.5+ -Xmx256 -jar MEME.jar"`-ról `"javaw.exe -version:1.5+ -Xmx<desired amount> -jar MEME.jar"`-ra .

8.3 Help menü

Ez a kézikönyv és a MEME Tutorial elérhető a *Help* menüből. Megjegyzendő, hogy ezen pdf dokumentumok megnyitásához Acrobat Reader, vagy más pdf-megjelenítő program szükséges. Az Acrobat Reader letölthető a következő helyről:

<http://www.adobe.com/products/acrobat/readstep2.html>

8.4 A művelet állását jelző ablak (progress window)

A művelet állását jelző ablakon a hátralévő idő (time left) a ténylegesen hátralévő idő felső becslése. Általában, az eleinte mutatott idő több, mint amennyi idő alatt a művelet ténylegesen lefut, kivéve ha a MEME számára rendelkezésre álló memória mérete, vagy a CPU idő futás közben ingadozik. Utóbbi esetben a becslés nem iránymutató: időről időre nőlni, illetve csökkenhet az értéke a fent leírt körülményektől függően.

9 Zárzó

A MEME egy olyan eszköz, ami szimulációk batch jellegű futtatására és a generált adatok feldolgozására való. Lehetőséget kínál arra, hogy a felhasználó adatbázis(ok)ban tárolja, és kezelje a nyers adatokat, és hogy olyan számított táblákat hozzon létre, melyek grafikonokon ábrázolhatók.

A közeljövőben várható, hogy a MEME képes lesz MASS/Repast szimulációk közvetlen futtatására, és eredményeik belső összegyűjtésére, és arra, hogy együttműködjön egyéb, már létező statisztikai szoftverekkel, mint az R és a Matlab. Ezek a fejlesztések tovább fogják egyszerűsíteni a modellezési munkát, ezáltal a felhasználók magukra a modellekre koncentrálhatnak.

10 Referenciák

- [1] MASS – Multi-Agent Simulation Suite. © AITIA Internation Inc.
http://www.aitia.ai/services_and_products/simulation_systems/mass
- [2] Repast - Recursive Porus Agent Simulation Toolkit
<http://repast.sourceforge.net/>
- [3] R - The R Project for Statistical Computing
<http://www.r-project.org/>
- [4] Matlab
<http://www.mathworks.com/>
- [5] Beanshell Scripting
<http://www.beanshell.org/>