
금융시스템 시뮬레이션

국제결제은행(BIS)

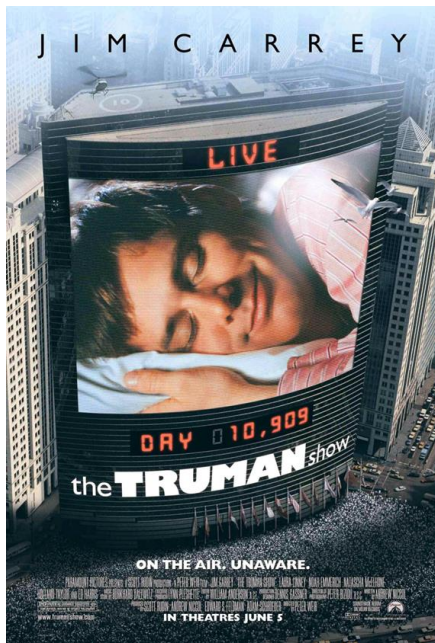
권병천

개요

- 1부 - 소개
 - 시뮬레이션
 - 사례
- 2부 - 은행 시스템 시뮬레이터
 - 구축
 - 실험

시뮬레이션

모델을 통해서 수행되는 동작



$$\begin{aligned}\tilde{y}_t &= -\frac{1}{\sigma}(i_t - E_t\{\pi_{t+1}\} - r_t^n) + E_t\{\tilde{y}_{t+1}\} \\ \pi_t &= \beta E_t\{\pi_{t+1}\} + \kappa \tilde{y}_t \\ i_t &= \rho + \gamma i_{t-1} + \phi_\pi \pi_t + \phi_y \tilde{y}_t + v_t \\ r_t^n &= \rho + \sigma \psi_{ya}^n E_t\{\Delta a_{t+1}\} \\ c_t &= E_t\{c_{t+1}\} - \frac{1}{\sigma}(i_t - E_t\{\pi_{t+1}\} - \rho) \\ &\quad + \frac{1}{\sigma}(1 - \rho_z)z_t\end{aligned}$$



시뮬레이션

- 구체적 모델 (concrete)
 - 구체적인 물리적 형태를 지닌 모델 (ex. 모델하우스)
- 수리모델 (mathematical)
 - 어떤 현상을 수학의 수식을 통해 재현한 모델 (ex. 계량경제모형)
- 계산 모델 (computational)
 - 컴퓨터 계산(규칙, 조건) 을 통해 재현한 모델 (ex. 행위자기반모형)

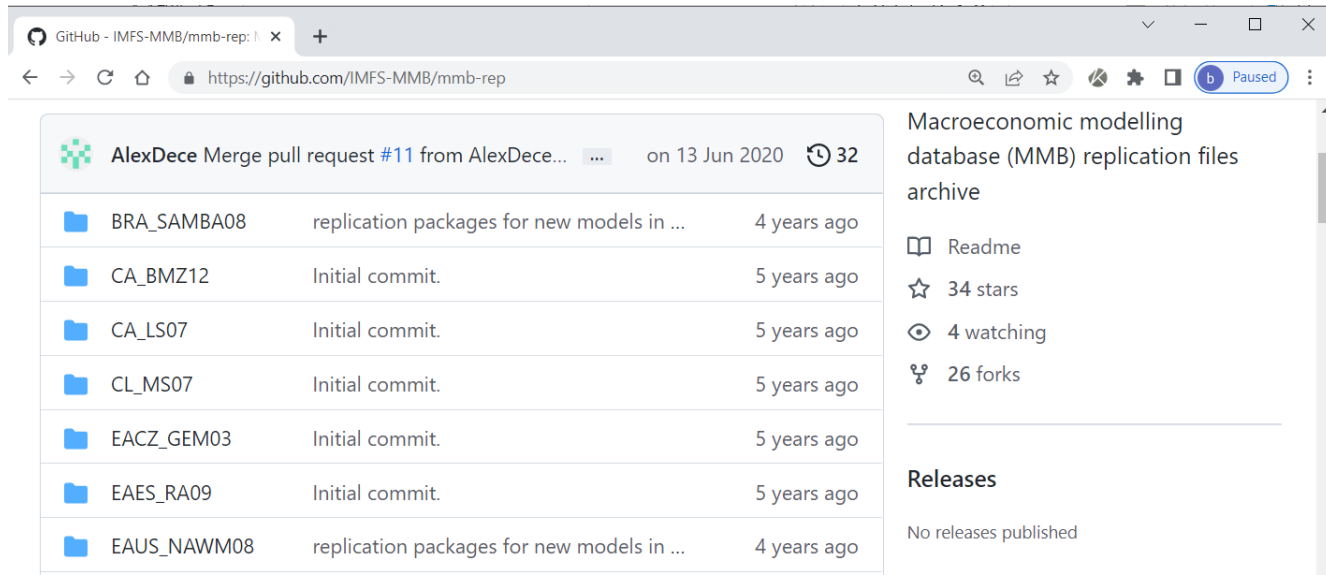
	시간	비용	정확도
구체적			
수리			
계산			

시뮬레이션 - 구체적 모형



시뮬레이션 – 수리적 모형

- 거시경제 모형 데이터베이스 ([MMB; Macroeconomic Model Data Base](https://github.com/IMFS-MMB/mmb-rep))
 - 150 여개 구조적 거시경제모형의 결과를 비교분석할 수 있는 서비스



GitHub - IMFS-MMB/mmb-rep: Macroeconomic modelling database (MMB) replication files archive

on 13 Jun 2020 32

Folder Name	Description	Commit Time
BRA_SAMBA08	replication packages for new models in ...	4 years ago
CA_BMZ12	Initial commit.	5 years ago
CA_LS07	Initial commit.	5 years ago
CL_MS07	Initial commit.	5 years ago
EACZ_GEM03	Initial commit.	5 years ago
EAES_RA09	Initial commit.	5 years ago
EAUS_NAWM08	replication packages for new models in ...	4 years ago

Readme

34 stars

4 watching

26 forks

Releases

No releases published

1.1 NK_AFL15:Angeloni et al. (2015)

Angeloni et al. (2015) propose a DSGE model where banks are subject to runs, modelled as a discipline device in the spirit of Diamond and Rajan (2000, 2001). Banks invest in risky projects and fund operations via bank equity and run-prone deposits, such that their funding structure endogenously determines bank fragility. In the model, expansion-

```
stst2=fsolve(@(stst2) (1-BETTA*(stst2+HH)/(2-BET+CR*(1+BET)))-THETA*((1-BETTA*(stst2+HH)/(2-BET+CR*(1+BET))))
rass=stst2;
qss=1;
zss=rass-(1-DELTA);
rokss=zss+(1-DELTA);
rnss=PAIss/BETTA;
mcss=(EPSI-1)/EPSI;
YoK=zss/(mcss*ALFA);
IoK=DELTA;
IoY=IoK/YoK;
CoY=1-GY-IoY;
```

$$L_t = (1 - \xi)(Q_{t-1}K_t - NW_t) = D_t + BK_t \tag{12}$$

The liability structure of the bank, measured by the deposit share, $d_t = \frac{D_t}{L_t}$ ¹⁸, is determined by a bank liability manager on behalf of the external financiers (depositors and bank capitalists). What we call, for terminological simplicity, deposits are not traditional retail deposits, which usually are nearly fully insured; they are uninsured short term funding instruments

Macroeconomic Model Database

Online Comparison Platform

Welcome to the Macroeconomic Model Database (MMB) v.3.1. Contribute your model using our [Contribute Form](#) or get in touch via our [forum](#).

Models (3/151) [Clear](#)

Calibrated	Estimated US	Estimated Euro Area	Other
<input checked="" type="checkbox"/> NK_AFL15	<input type="checkbox"/> US_RE09	<input type="checkbox"/> EA_ALSV06	<input type="checkbox"/> BRA_SAMBA08
<input checked="" type="checkbox"/> NK_BGEU10	<input type="checkbox"/> US_RS99	<input type="checkbox"/> EA_AWM05	<input type="checkbox"/> CA_BMZ12
<input checked="" type="checkbox"/> NK_BGG99	<input type="checkbox"/> US_SW07	<input type="checkbox"/> EA_BE15	<input type="checkbox"/> CA_LS07
<input type="checkbox"/> NK_BGUS10	<input type="checkbox"/> US_VI16bgg	<input type="checkbox"/> EA_BF17	<input type="checkbox"/> CA_ToTEM10
<input type="checkbox"/> NK_CFP10	<input type="checkbox"/> US_VI16gk	<input type="checkbox"/> EA_CKL09	<input type="checkbox"/> CL_MS07
<input type="checkbox"/> NK_CGG02	<input type="checkbox"/> US_VMDno	<input type="checkbox"/> EA_CW05fm	<input type="checkbox"/> EACZ_GEM03
<input type="checkbox"/> NK_CGG99	<input type="checkbox"/> US_VMDop	<input type="checkbox"/> EA_CW05ta	<input type="checkbox"/> EAES_RA09
<input type="checkbox"/> NK_CK08	<input type="checkbox"/> US_YR13	<input type="checkbox"/> EA_DKR11	<input type="checkbox"/> EAUS_NAWM08

[? Documentation of Models](#)

Policy Rules (1/9) [Clear](#)

- ☐ Christiano et al. (2014)
- ☐ Coenen et al. (2012)
- ☐ Gerdesmeier & Roffia (2004)
- ☐ Levin et al. (2003)
- ☐ Orphanides & Wieland (2008)
- ☐ Orphanides & Wieland (2013)
- ☐ Smets & Wouters (2007)
- ☒ Taylor (1993)

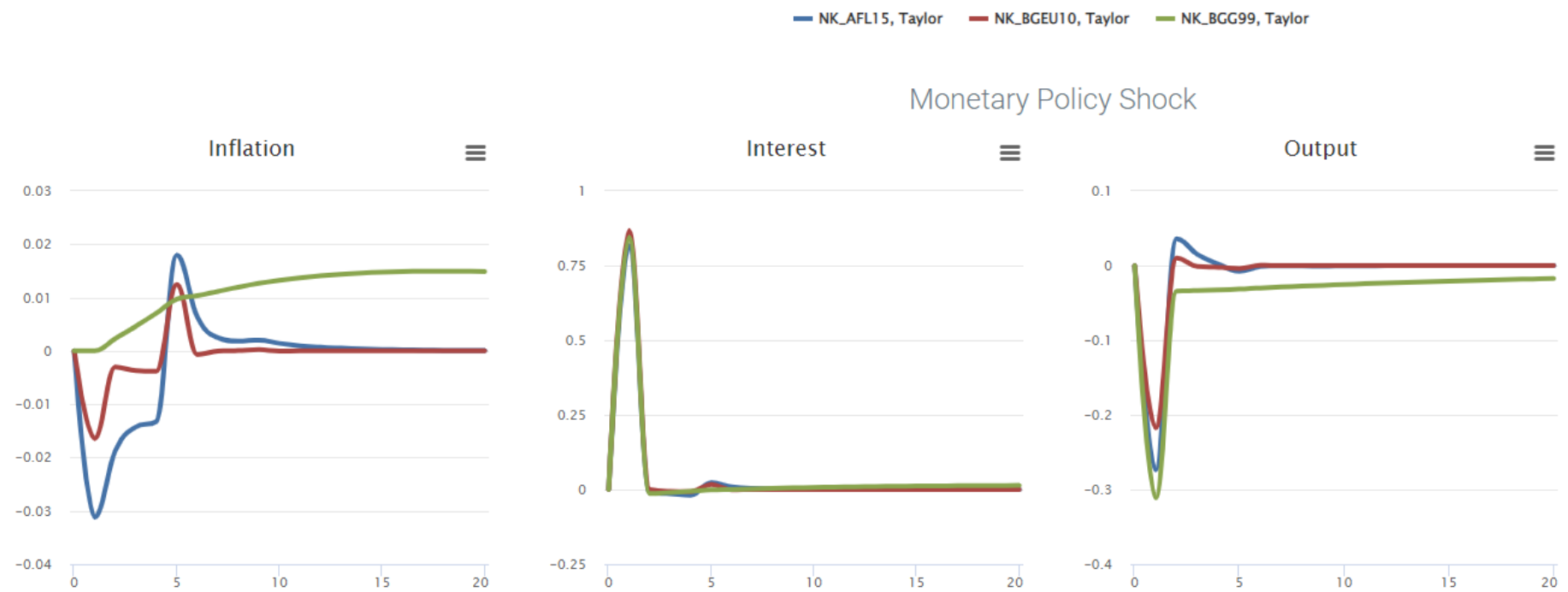
Shocks (1/2) [Select all](#) [Clear](#)

- ☒ Monetary Policy Shock
- ☐ Fiscal Policy Shock

Variables (4/4) [Select all](#) [Clear](#)

- ☒ Inflation
- ☒ Interest
- ☒ Output
- ☒ Output Gap

Comparison (3 Models, 1 Policy Rule, 1 Shock)





Online Comparison Platform

You can explore model comparison techniques on the [Online Comparison Platform \(OCP\)](#). It offers many of the same features as MMB 3.1, and gives you access to pre-run simulations of all models for a subset of shocks and policy rules. For some examples of interesting comparisons please check out our [Blog](#).

Download

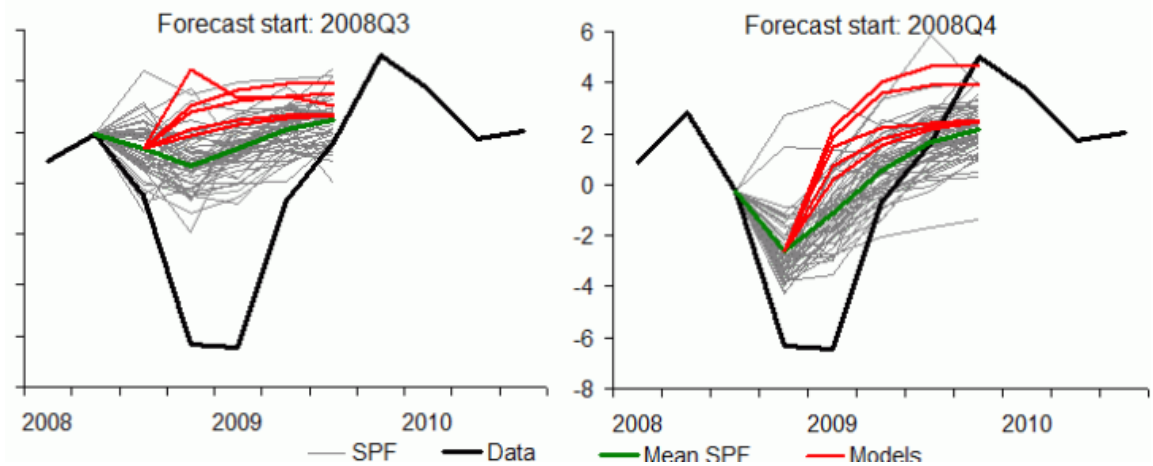
The latest MMB versions 2.3, 3.0 and 3.1 are released under the auspices of the [Macroeconomic Model Comparison Initiative \(MMCI\)](#), a joint project of the [Hoover Institution](#) at Stanford University and the [Institute for Monetary and Financial Stability \(IMFS\)](#) at Goethe University Frankfurt that is supported financially by the [Alfred P. Sloan Foundation](#).

[Download MMB 3.1 \(Windows\)](#)[Download MMB 3.1 \(Mac OS\)](#)[Download MMB 3.1 \(Linux\)](#)

For windows users we provide an "all-in-one installer", which you can download here. It installs the latest version of the MMB together with a suitable combination of Dynare and Octave.

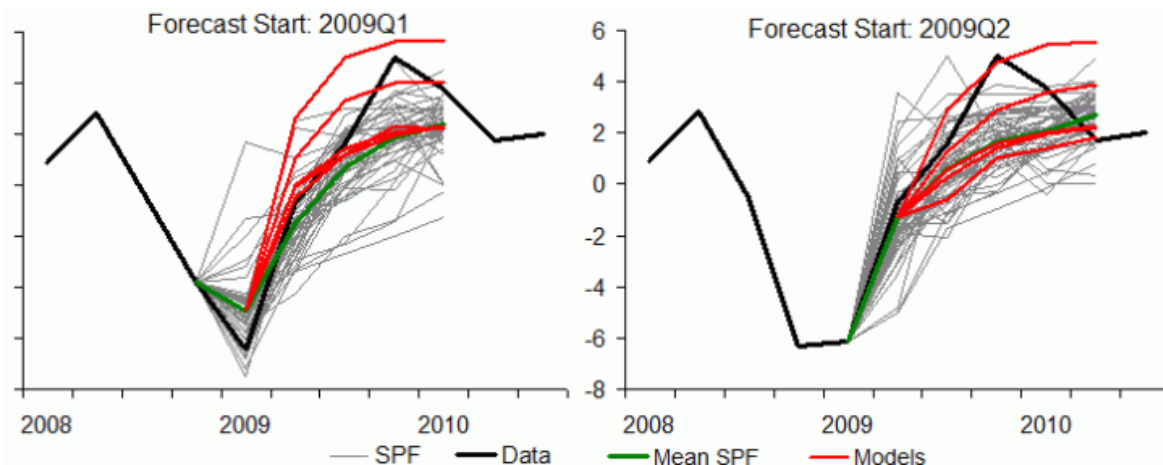
[Download All-in-one installer \(Windows\)](#)

(참고): 2008년 금융위기에 대한 거시경제 모형 - 전문가 패널 GDP 성장률 전망



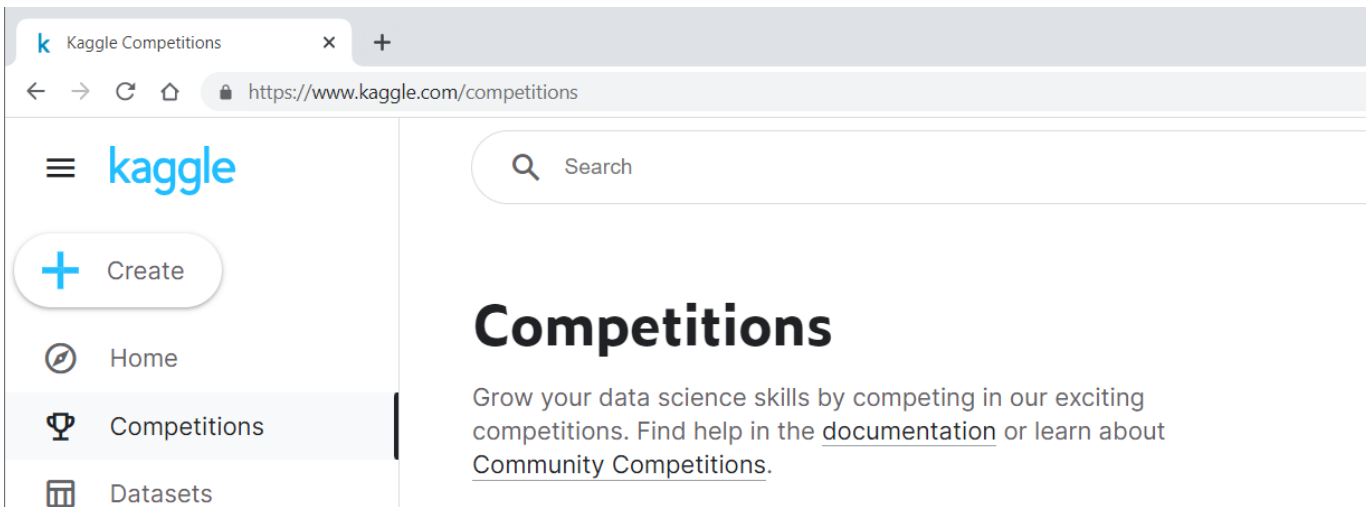
<https://cepr.org/voxeu/columns/macroeconomic-model-comparisons-and-forecast-competitions>

* SPF: Survey of Professional Forecasters



시뮬레이션 - 수리적 모형

- 머신 러닝 - 훈련 데이터를 이용해 수리적 모형으로 학습을 통한 예측



사례: 뉴욕시 택시요금 예측

<https://www.kaggle.com/competitions/new-york-city-taxi-fare-prediction>

1,483 1,566 20,088
Teams Competitors Entries



key	pickup_date	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	# passenger...
2015-01-27 13:08:24.000000 2	2015-01-27 13:08:24 UTC	-73.97332000732 4219	40.763805389404 3	-73.98143005371 0938	40.743835449218 75	1
2015-01-27 13:08:24.000000 3	2015-01-27 13:08:24 UTC	-73.98686218261 7188	40.719383239746 094	-73.99888610839 8438	40.739200592041 016	1

New York City Taxi Fare Prediction

Cleansing+EDA+Modelling(LGBM + XGBoost starters)

Notebook Data Logs Comments (64)

115]:

```
Index(['pickup_longitude', 'pickup_latitude', 'dropoff_longitude',  
      'dropoff_latitude', 'passenger_count', 'H_Distance', 'Year', 'Month',  
      'Date', 'Day of Week', 'Hour'],  
      dtype='object')
```

```
import xgboost as xgb
```

```
dtrain = xgb.DMatrix(x_train, label=y_train)  
dtest = xgb.DMatrix(x_test)
```

<https://www.kaggle.com/code/madhurisivalenka/cleansing-eda-modelling-lgbm-xgboost-starters>

1. Initialize model with a constant value:

$$\hat{f}_{(0)}(x) = \arg \min_{\theta} \sum_{i=1}^N L(y_i, \theta).$$

2. For $m = 1$ to M :

1. Compute the 'gradients' and 'hessians':

$$\hat{g}_m(x_i) = \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)} \right]_{f(x)=\hat{f}_{(m-1)}(x)}.$$
$$\hat{h}_m(x_i) = \left[\frac{\partial^2 L(y_i, f(x_i))}{\partial f(x_i)^2} \right]_{f(x)=\hat{f}_{(m-1)}(x)}.$$

2. Fit a base learner (or weak learner, e.g. tree) using the training set $\left\{ x_i, -\frac{\hat{g}_m(x_i)}{\hat{h}_m(x_i)} \right\}_{i=1}^N$

$$\hat{\phi}_m = \arg \min_{\phi \in \Phi} \sum_{i=1}^N \frac{1}{2} \hat{h}_m(x_i) \left[-\frac{\hat{g}_m(x_i)}{\hat{h}_m(x_i)} - \phi(x_i) \right]^2.$$

$$\hat{f}_m(x) = \alpha \hat{\phi}_m(x).$$

3. Update the model:

$$\hat{f}_{(m)}(x) = \hat{f}_{(m-1)}(x) + \hat{f}_m(x).$$

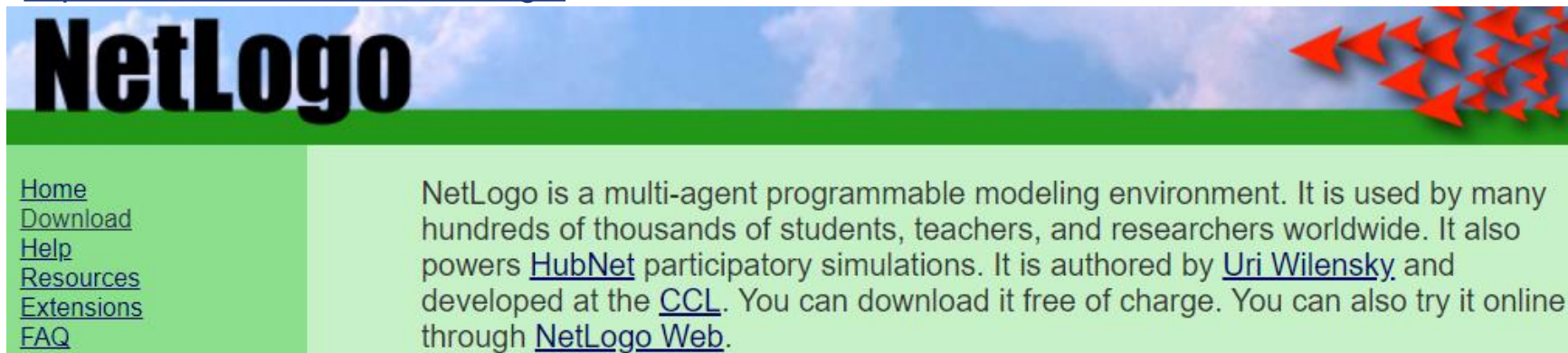
3. Output $\hat{f}(x) = \hat{f}_{(M)}(x) = \sum_{m=0}^M \hat{f}_m(x).$

<https://en.wikipedia.org/wiki/XGBoost>

시뮬레이션 - 계산 모형

- Heterogeneous - 다양한 특성을 가진 행위자 구현
- Dynamics - 시간 변화에 따른 파라미터 값 최적화
- Bandwagon effect - 주변행위자의 영향을 받아 발생하는 편승효과 구현

<https://ccl.northwestern.edu/netlogo/>



시뮬레이션 - 계산 모형

- 상향식 (Bottom-up) 접근

powered by NetLogo



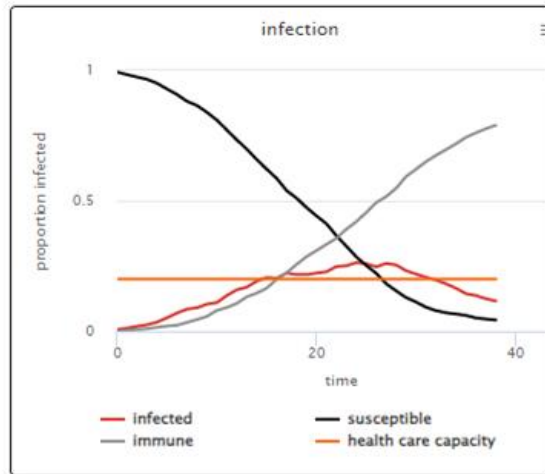
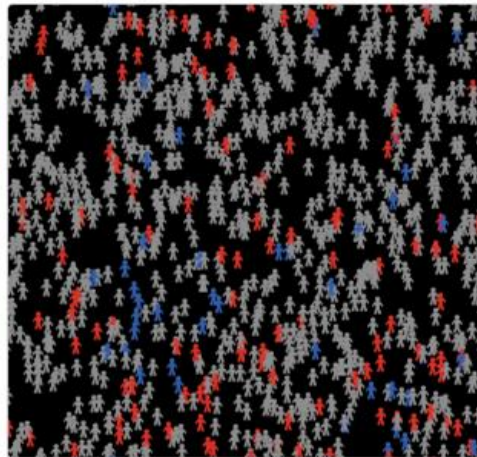
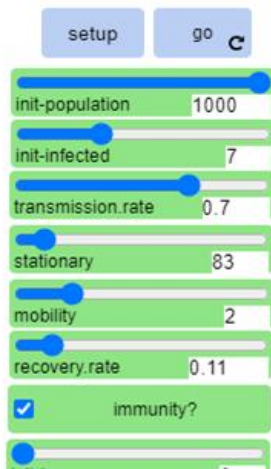
Mode: Interactive

Commands and Code: Bottom

COVID-19 VIRUS SPREAD

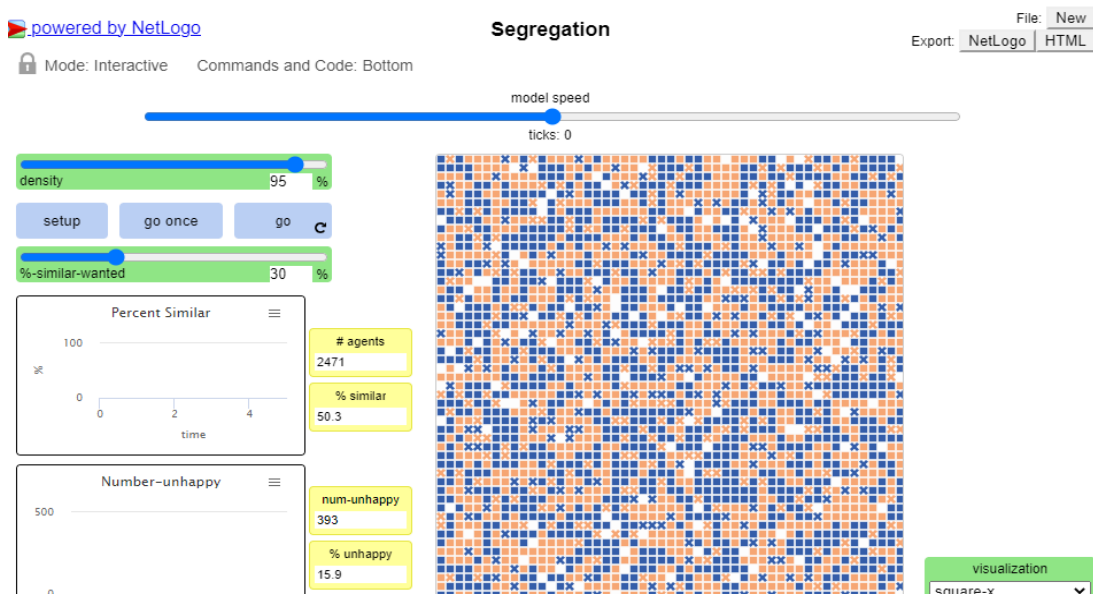
model speed

ticks: 38



시뮬레이션 사례 - 인종 분리 모형

- 토머스 셸링 - 1970년대 초반 미국 도시에서 나타난 인종 분리의 경향 - 인종주의?



데모 사이트: [넷로고](#)

Mode: Interactive Commands and Code: Bottom

model speed

ticks: 34

density 95 %

setup

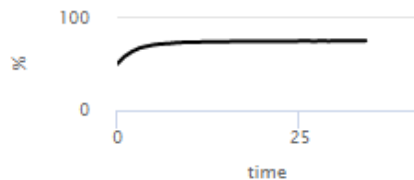
go once

go



%-similar-wanted 30 %

Percent Similar



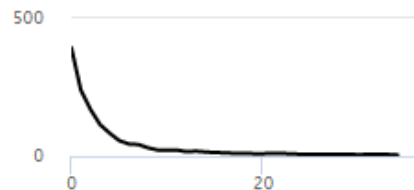
agents

2471

% similar

75.6

Number-unhappy

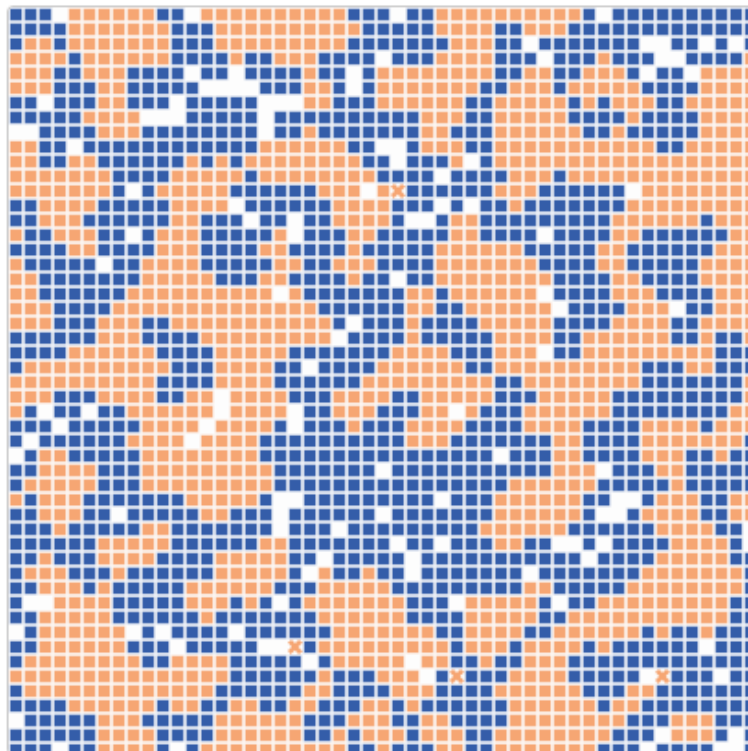


num-unhappy

4

% unhappy

0.2



visualization

square-x



Recycle Bin



금융시스템
사물레이...



금융시스템
사물레이...



금융강좌



Type here to search



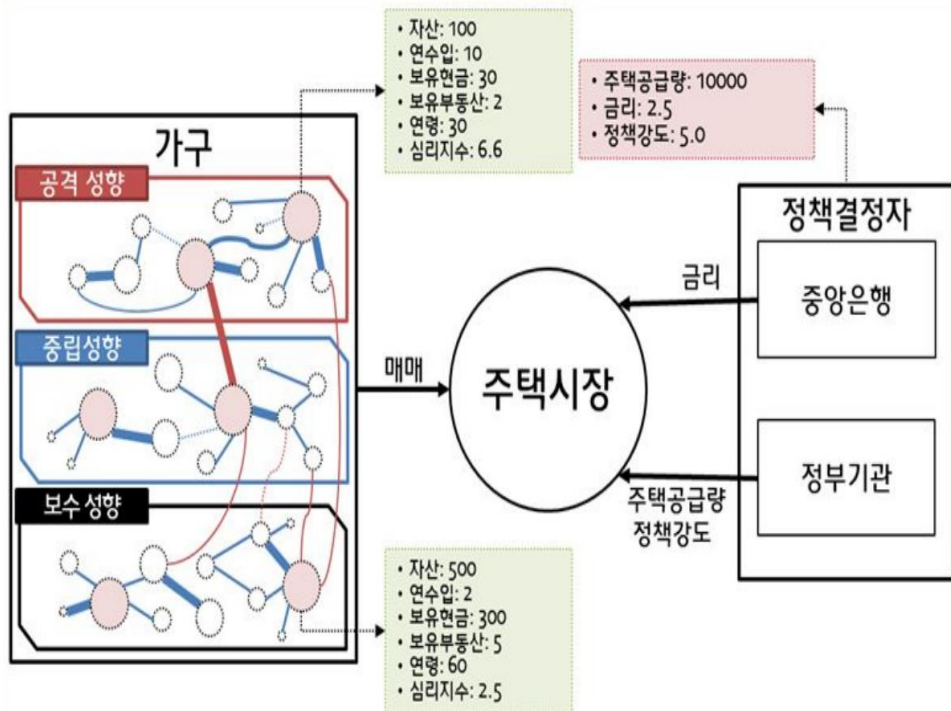
74°F Cloudy



11:02 AM
9/16/2022



시뮬레이션 사례 – 주택매매 시장 모형



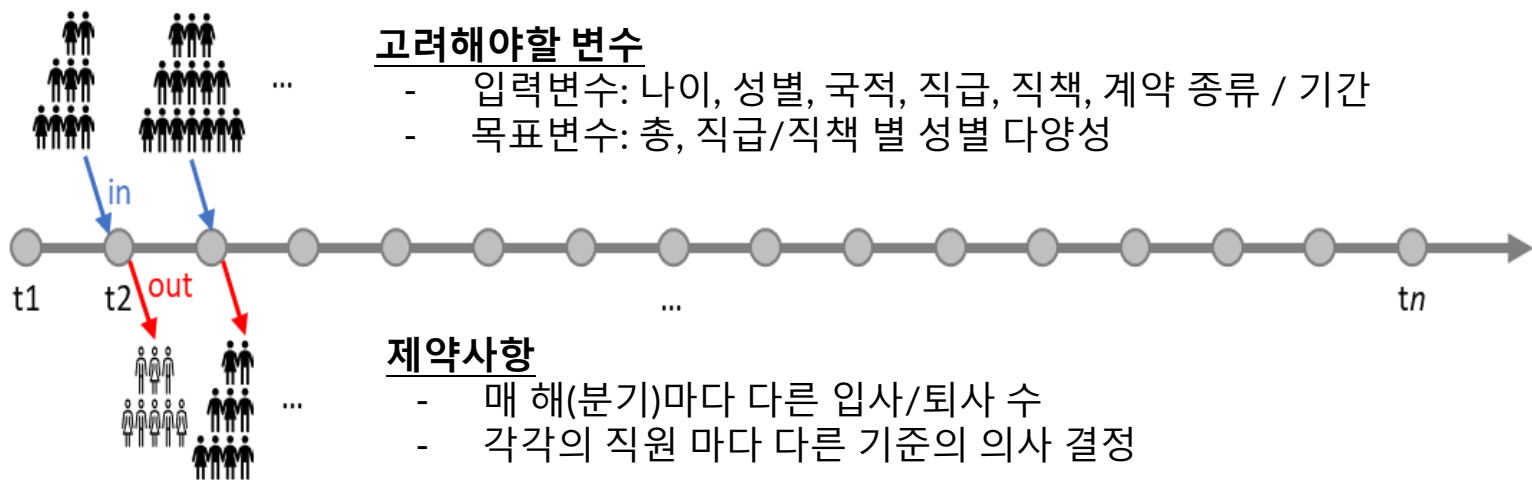
행위자: 가구, 정부, 중앙은행

시중 금리, 주택 공급량에 따라 가구 행위자는 다른 판단 기준을 가지고 주택매매 결정

→ 가격의 급격한 쓸림을 구현할 수 있는가?

시뮬레이션 사례 – 인사/채용 전망 시뮬레이터

- 인적자원의 다양성을 반영한 실제적 분석 가능
- 과거 인사데이터를 기반으로, 모형에서 사용하는 파라미터 값 최적화



시뮬레이션 사례 – 인사/채용 전망 시뮬레이터

구현: <https://colab.research.google.com/drive/1eQNaZs-0Siw6LiHtkv59ZCaCbVGTboc3?usp=sharing>

	성별	국적	생년월일	직급	계약형태	직책	부서명	입사일	종료일
0	여	한국	1980-01-01	가급	정규직	사원	연구분석부	2016-05-15	NaT
1	여	한국	1980-01-01	라급	정규직	사원	금융연구부	2013-09-30	NaT
2	여	한국	1970-01-01	라급	계약직	사원	재산관리부	2012-05-08	2024-04-22

- 정년퇴직: 61
- 계약직 전환 비율: 20%
- 계약직 연장 연한: 3 years
- 여성 채용비율: 50%
- 계약직 사직 비율: 2%
- 정규직 사직 비율: 1%

```
new_staff_female_ratio=0.7 # 여성 채용 비율
)
```

```
sdmodel.run_model()
```

```
Date: 2024-01, #Staff: 100, #남: 58, #여: 42
Date: 2025-01, #Staff: 100, #남: 57, #여: 43
Date: 2026-01, #Staff: 100, #남: 60, #여: 40
Date: 2027-01, #Staff: 100, #남: 61, #여: 39
Date: 2028-01, #Staff: 100, #남: 59, #여: 41
Date: 2029-01, #Staff: 100, #남: 58, #여: 42
Date: 2030-01, #Staff: 100, #남: 57, #여: 43
Date: 2031-01, #Staff: 100, #남: 56, #여: 44
Date: 2032-01, #Staff: 100, #남: 37, #여: 63
```

인사채용 시뮬레이션.ipynb - Colab

https://colab.research.google.com/drive/1eQNaZs-0Siw6LiHtkv59ZCaCbVGTboc3#scrollTo=VIHp6xkzsenn

Comment

Share

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM 100% Disk 100% Editing

8s

!pip install mesa

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>

Collecting mesa

Downloading Mesa-1.0.0-py3-none-any.whl (2.5 MB)

2.5 MB 15.0 MB/s

Requirement already satisfied: tornado in /usr/local/lib/python3.7/dist-packages (from mesa) (5.1.1)

Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages (from mesa) (1.21.6)

Requirement already satisfied: networkx in /usr/local/lib/python3.7/dist-packages (from mesa) (2.6.3)

Collecting cookiecutter

Downloading cookiecutter-2.1.1-py2.py3-none-any.whl (36 kB)

Requirement already satisfied: pandas in /usr/local/lib/python3.7/dist-packages (from mesa) (1.3.5)

Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist-packages (from mesa) (4.64.1)

Requirement already satisfied: click in /usr/local/lib/python3.7/dist-packages (from mesa) (7.1.2)

Requirement already satisfied: pyyaml>=5.3.1 in /usr/local/lib/python3.7/dist-packages (from cookiecutter->mesa) (6.0)

Collecting jinja2-time>=0.2.0

Downloading jinja2_time-0.2.0-py2.py3-none-any.whl (6.4 kB)

Requirement already satisfied: Jinja2<4.0.0,>=2.7 in /usr/local/lib/python3.7/dist-packages (from cookiecutter->mesa) (2.11.3)

Requirement already satisfied: requests>=2.23.0 in /usr/local/lib/python3.7/dist-packages (from cookiecutter->mesa) (2.23.0)

Collecting binaryornot>=0.4.4

Downloading binaryornot-0.4.4-py2.py3-none-any.whl (9.0 kB)

Requirement already satisfied: python-slugify>=4.0.0 in /usr/local/lib/python3.7/dist-packages (from cookiecutter->mesa) (6.1.2)

Requirement already satisfied: chardet>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from binaryornot>=0.4.4->cookiecutter->mesa) (3.0.4)

Requirement already satisfied: MarkupSafe>=0.23 in /usr/local/lib/python3.7/dist-packages (from Jinja2<4.0.0,>=2.7->cookiecutter->mesa) (2.0.1)

Collecting arrow

8s

completed at 1:59 PM

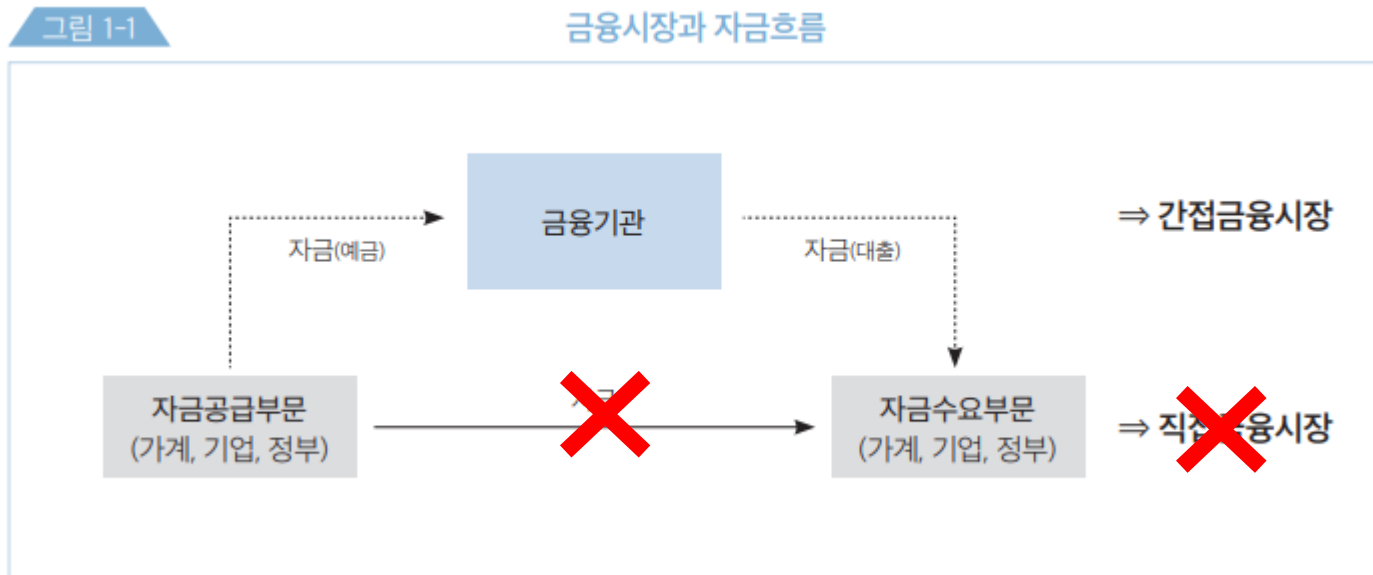
Type here to search

74°F Cloudy

2:00 PM 9/16/2022

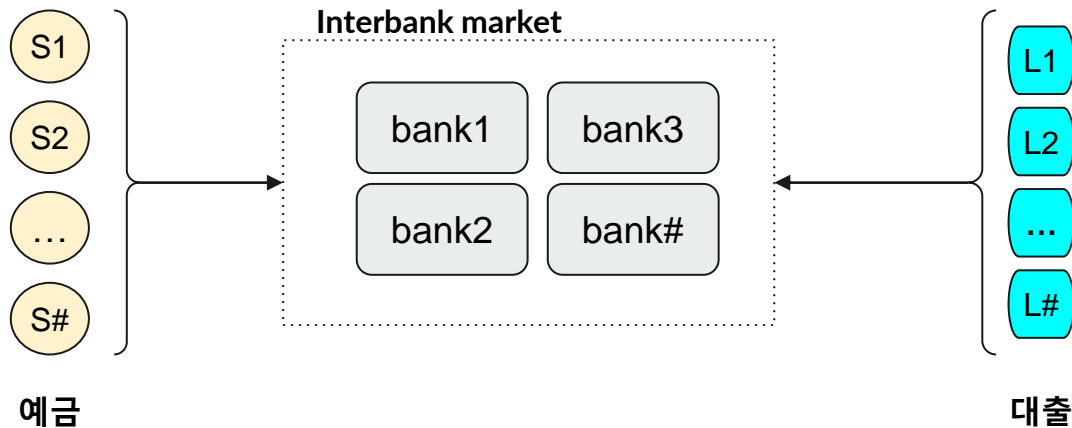
은행시스템 시뮬레이션

- 금융시장에서 발생하는 예금, 대출, 부실화, 파급효과, 자금인출을 시뮬레이션으로 구현



은행시스템 시뮬레이션

- 행위자 - 은행 (Bank), 예금 (Saver), 대출 (Loan)



은행시스템 시뮬레이션

- 은행 행위 설정
 - 금융감독 규제 충족
 - 지급준비율 (MRR: Minimum Reserve Ratio)
 - 자기자본비율 (CAR: Capital Adequacy Ratio)
 - 대출 부실화 대비 - 대손충당금 (Provision) 설정
 - 지불능력 (Solvency) 검증
 - $\text{자본} = \text{전기 자본} + \text{순이자 수입} - \text{대출부실 손실} + \text{대손충당 증감}$
 - 지급불능 (Insolvency) 시, 대출 유동화 (liquidated) 실행 → 헐값매각 (fire sale)
 - 타 은행 부실화에 따른 2차 파급효과 계산 (Second round effects)
 - 배당지급, 대출 확장, 은행간 대출, 긴급 유동화 지원

은행시스템 시뮬레이션

- 예금 (Saver) 행위설정
 - 예금에 대한 이자수익 실현
 - 예치 은행 변경 결정
- 대출 (Loan) 행위설정
 - 대출 속성: 대출금액, 부도 확률, 위험 가중치, 부도시 대출 회수율, 할값 매각시 손실률
 - 대출 이자 (Loan rate)는 대출 속성에 따라 달라짐 (Fig 4)
 - 대출 은행에게 이자지급 및 연장. 은행 부실시 연장 거부
 - 대출이 부실화 될 경우, 부도처리 및 대출 회수율 만큼 은행 상환

시뮬레이션 데모

- Case 1: 시뮬레이션 개념 이해 - 행위자, 공간, 시간
- Case 2: 다수 은행, 랜덤 워크
- Case 3: 단일 은행, 다수 예금
- Case 4: 단일 은행, 예금, 대출
- Case 5: 단일 은행, 예금, 대출, 이자율
- Case 6: 단일 은행, 예금, 대출, 대출 부실화,뱅크런

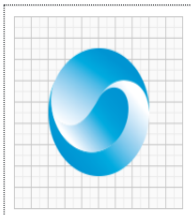
복잡도 증가



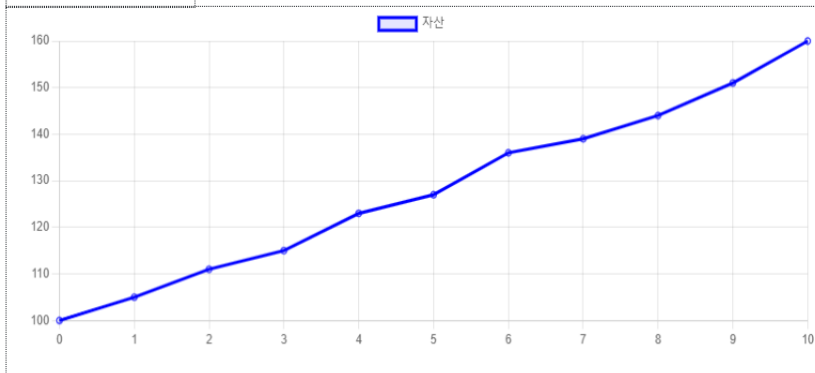
시뮬레이션 데모: Case 1 – 단일 은행

- 개념이해

- 행위자 정의
- 공간
- 시간에 대한 개념
- 결과 해석



```
class Bank(mesa.Agent):  
    def __init__(self, unique_id, model):  
        super().__init__(unique_id, model)  
        self.asset = 100  
  
    def step(self):  
        self.asset += np.random.randint(0, 10)
```





case1_onebank.py U X

notebook > case1_onebank.py >...

```
1  """
2  사례 1 - 단일 뱅크, 랜덤워크
3  """
4
5  import mesa
6  import numpy as np
7
8  from mesa.visualization.UserParam import UserSettableParameter
9
10
11 class Bank(mesa.Agent):
12     def __init__(self, unique_id, model):
13         super().__init__(unique_id, model)
14         self.asset = 100
15
16     def step(self):
17         self.asset += np.random.randint(0, 10)
18
19
20 def compute_asset(model):
21     return sum([agent.asset for agent in model.schedule.agents])
22
23
24 class ModelBankingSystem(mesa.Model):
25     def __init__(self, N, width, height, num_step):
26         self.num_agents = N
27         self.num_steps = num_step
28         self.grid = mesa.space.MultiGrid(width, height, True)
29         self.schedule = mesa.time.RandomActivation(self)
30         self.datacollector = mesa.DataCollector(model_reporters={"자산": compute_asset})
```

PROBLEMS 1 TERMINAL

cmd + v [Icons]

(env) C:\Users\byeun\workspace\mesa\notebook>

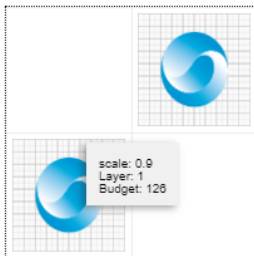


시뮬레이션 데모: Case 2 - 다수 은행, 랜덤 워크

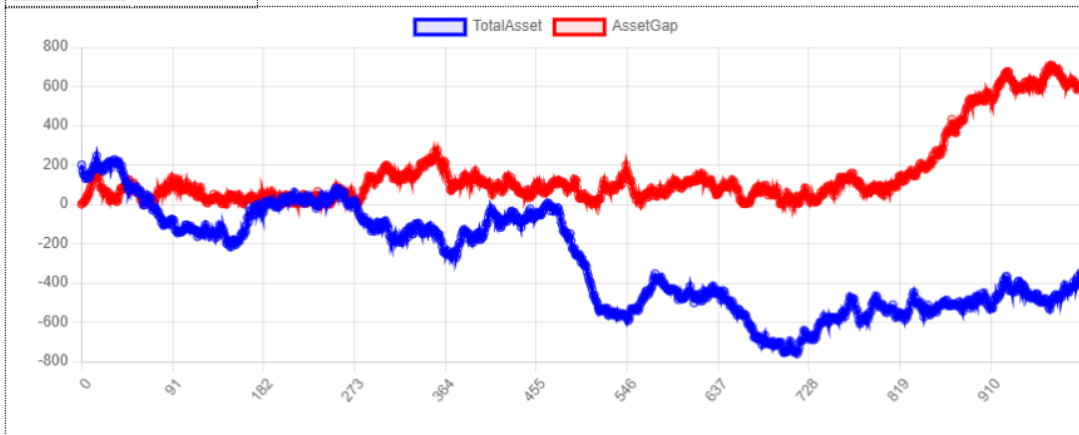
- 개념 이해

- 다수 행위자
- 랜덤 워크

Current Step: 1001



- 스텝: 1000
- 스텝별: 랜덤 값 (-20, 20) 생성 후 자산 추가





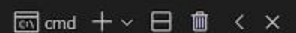
case2_twobanks.py U X



notebook > case2_twobanks.py > ...

```
1 """
2 사례 2 - 멀티 뱅크, 랜덤워크
3 """
4
5 import mesa
6 import numpy as np
7
8 from mesa.visualization.UserParam import UserSettableParameter
9
10
11 class Bank(mesa.Agent):
12     def __init__(self, unique_id, model):
13         super().__init__(unique_id, model)
14         self.asset = 100
15
16     def step(self):
17         self.asset += np.random.randint(-20, 20)
18
19
20 def compute_gap(model):
21     arr_asset = [agent.asset for agent in model.schedule.agents]
22
23     gap_value = np.max(arr_asset) - np.min(arr_asset)
24     print('asset_gap', gap_value)
25     # return np.sum(gap)
26     return int(gap_value)
27
28
29 def compute_asset(model):
30     return sum([agent.asset for agent in model.schedule.agents])
31
```

PROBLEMS 1 TERMINAL



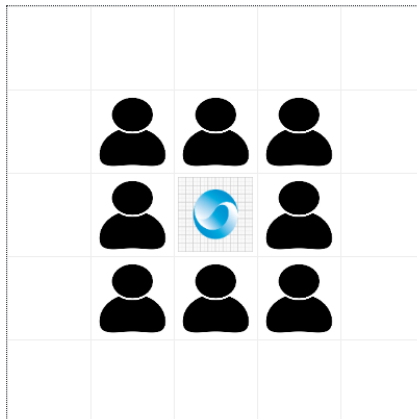
(env) C:\Users\byeun\workspace\mesa\notebook>



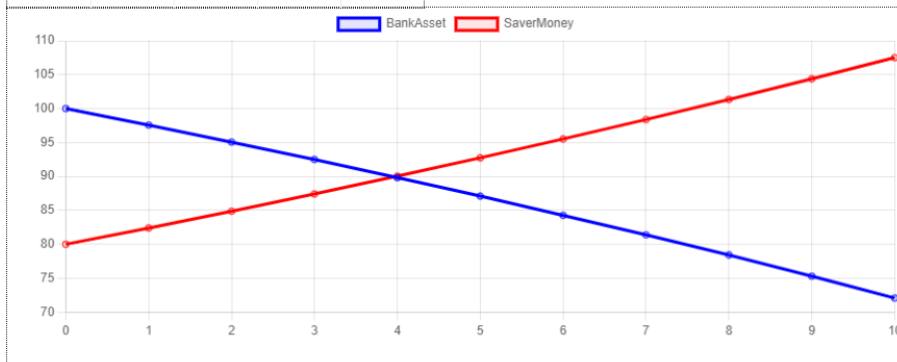
시뮬레이션 데모: Case 3 – 단일 은행, 다수 예금

- 개념 이해

- 다중 행위자



- 스텝: 10
- 스텝별: 은행이 예금자에게 3% 이자지급



case3_onebank_multisavers.py U X

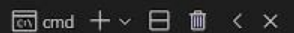


notebook > case3_onebank_multisavers.py > Saver > __init__ > model

```
8 from mesa.visualization.UserParam import UserSettableParameter
9
10
11 class Saver(mesa.Agent):
12     def __init__(self, unique_id, pos, model, money):
13         super().__init__(unique_id, pos)
14         self.money = money
15
16     def step(self):
17         self.money += self.money * 0.03
18
19
20 class Bank(mesa.Agent):
21     def __init__(self, unique_id, pos, model):
22         super().__init__(unique_id, pos)
23         self.model = model
24         self.asset = 100
25
26     def step(self):
27         for saver in self.model.schedule.agents:
28             if type(saver) is Saver:
29                 self.asset -= saver.money * 0.03
30
31
32 def compute_saver_money(model):
33     return sum([agent.money for agent in model.schedule.agents if type(agent) is Saver])
34
35
36 def compute_bank_asset(model):
37     return sum([agent.asset for agent in model.schedule.agents if type(agent) is Bank])
```

PROBLEMS 1

TERMINAL ...

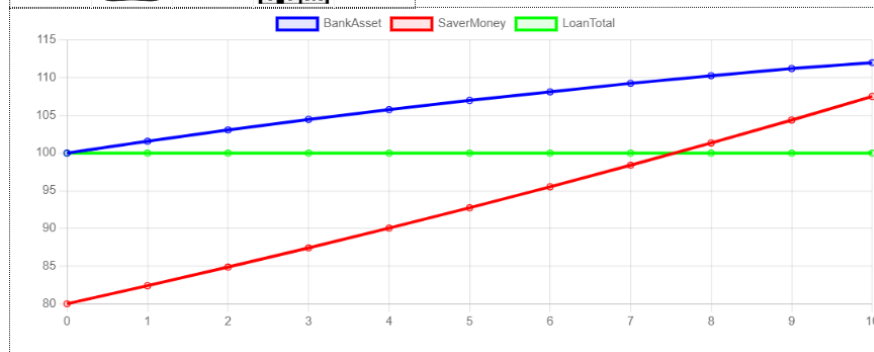
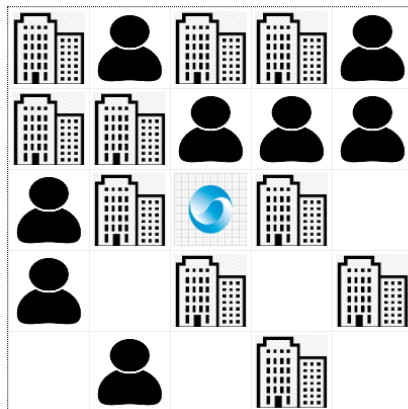


(env) C:\Users\byeun\workspace\mesa\notebook>



시뮬레이션 데모: Case 4 – 단일 은행, 예금, 대출

- 개념이해
 - 이질적 행위




```

11 class Saver(mesa.Agent):
12     def __init__(self, unique_id, pos, model, money):
13         super().__init__(unique_id, pos)
14         self.money = money
15
16     def step(self):
17         self.money += self.money * 0.03
18
19 class Loan(mesa.Agent):
20     def __init__(self, unique_id, pos, model, loan):
21         super().__init__(unique_id, pos)
22         self.loan = loan
23
24     def step(self):
25         pass
26
27 class Bank(mesa.Agent):
28     def __init__(self, unique_id, pos, model):
29         super().__init__(unique_id, pos)
30         self.model = model
31         self.asset = 100
32
33     def step(self):
34         for agent in self.model.schedule.agents:
35             if type(agent) is Saver:
36                 self.asset -= agent.money * 0.03
37             if type(agent) is Loan:
38                 self.asset += agent.loan * 0.04
39
40

```

TERMINAL ...

cmd + ⌘ + ⌃ + ⌫ < ×

```
(env) C:\Users\byeun\workspace\mesa\notebook>
```

시뮬레이션 데모: Case 5 – 단일 은행, 예금, 대출, 이자율

- 개념 이해

- 파라미터 값 설정

스텝수

10

예금자 수

20

대출 수

30

저축 이자율(%)

3

대출 이자율(%)

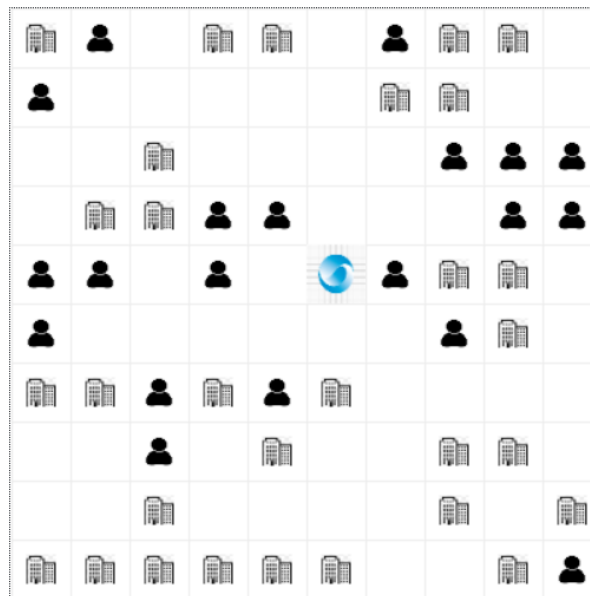
4

Frames Per Second

0

20

Current Step: 11





case5_onebank_multisavers_multiloans_flex.py U X

```
notebook > case5_onebank_multisavers_multiloans_flex.py > Loan > _init_  
1  """  
2  사례 5 - 단일 뱅크, 예금자(?) -이자(?%), 대출(?) -이자(?%)  
3  """  
4  
5  import mesa  
6  import numpy as np  
7  
8  from mesa.visualization.UserParam import UserSettableParameter  
9  
10  
11  class Saver(mesa.Agent):  
12      def __init__(self, unique_id, pos, model, money):  
13          super().__init__(unique_id, pos)  
14          self.money = money  
15          self.model = model  
16  
17      def step(self):  
18          self.money += self.money * self.model.saving_interest_rate  
19  
20  
21  class Loan(mesa.Agent):  
22      def __init__(self, unique_id, pos, model, loan):  
23          super().__init__(unique_id, pos)  
24          self.loan = loan  
25  
26      def step(self):  
27          pass  
28          # self.loan -= self.loan * 0.04  
29  
30
```

PROBLEMS 1 TERMINAL

```
(env) C:\Users\byeun\workspace\mesa\notebook>python case5_onebank_multisavers_multiloans_flex.py
```

시뮬레이션 데모: 종합예제

- 부도확률에 따른 최적 대출 이자율을 계산할 수 있을까?

스텝수

30

예금자 수

30

대출 수

30

초기 예금액

10

초기 대출액

10

저축 이자율(%)

3.5

대출 이자율(%)

4.5

부도율(%)

10.5

대출 부도비용에 따른 뱅크런 문턱값(%)

30

대출 부도에 따른 인출비율(%)

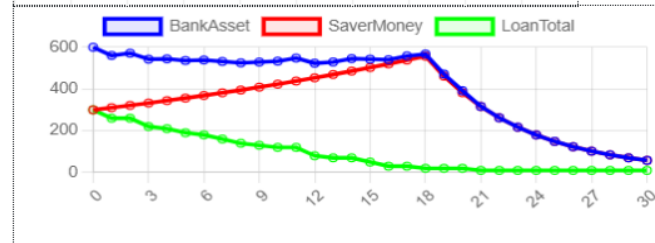
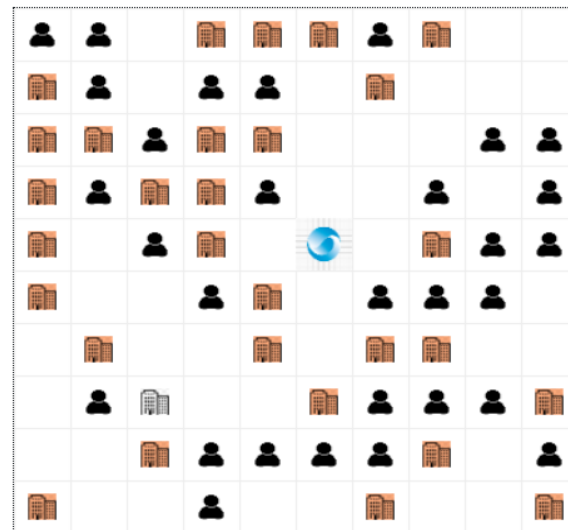
20

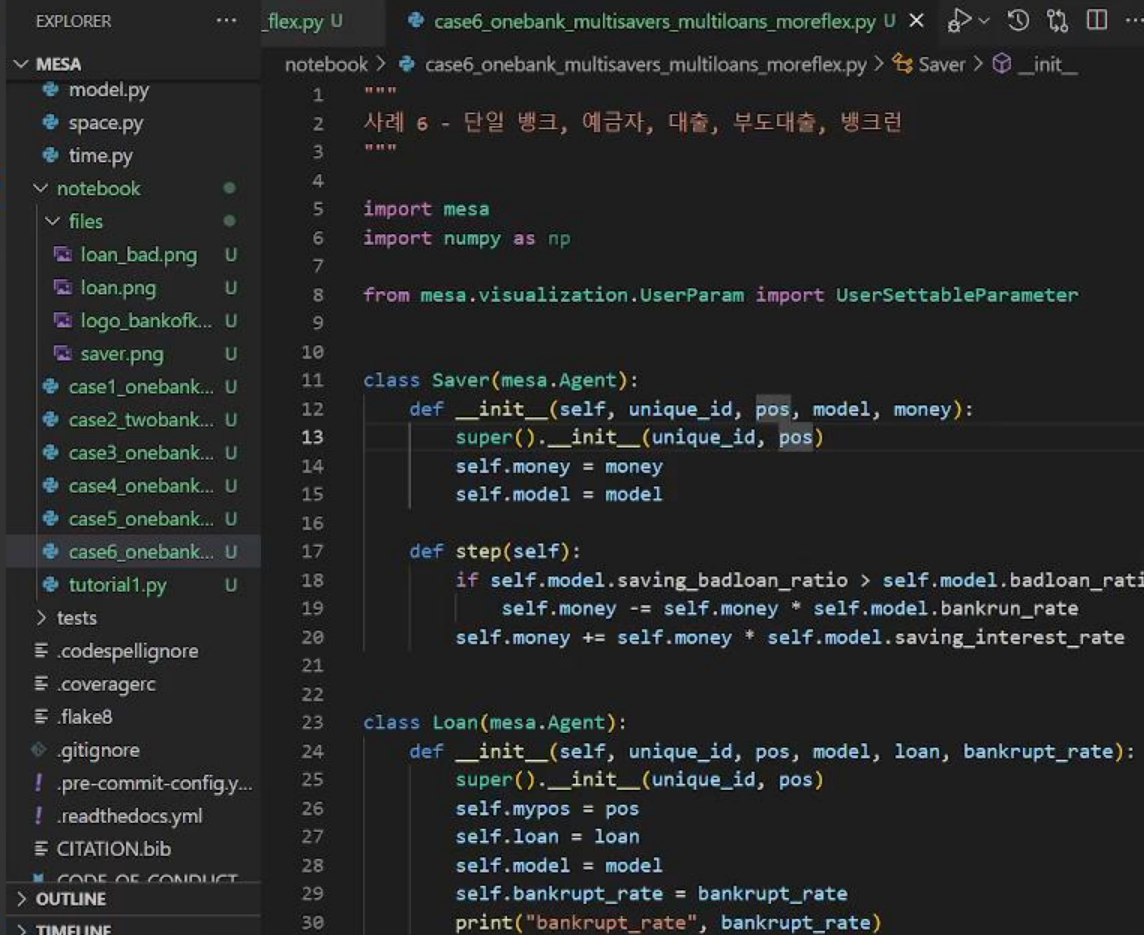
Frames Per Second

0

20

Current Step: 31





```
notebook > case6_onebank_multisavers_multiloans_moreflex.py > Saver > _init_
1  """
2  사례 6 - 단일 البنك, 예금자, 대출, 부도대출, 은행런
3  """
4
5  import mesa
6  import numpy as np
7
8  from mesa.visualization.UserParam import UserSettableParameter
9
10
11  class Saver(mesa.Agent):
12      def __init__(self, unique_id, pos, model, money):
13          super().__init__(unique_id, pos)
14          self.money = money
15          self.model = model
16
17      def step(self):
18          if self.model.saving_badloan_ratio > self.model.badloan_ratio:
19              self.money -= self.money * self.model.bankrun_rate
20              self.money += self.money * self.model.saving_interest_rate
21
22
23  class Loan(mesa.Agent):
24      def __init__(self, unique_id, pos, model, loan, bankrupt_rate):
25          super().__init__(unique_id, pos)
26          self.mypos = pos
27          self.loan = loan
28          self.model = model
29          self.bankrupt_rate = bankrupt_rate
30          print("bankrupt_rate", bankrupt_rate)
```

PROBLEMS1

TERMINAL

...

python

+

⌵

📄

🗑️

⏪

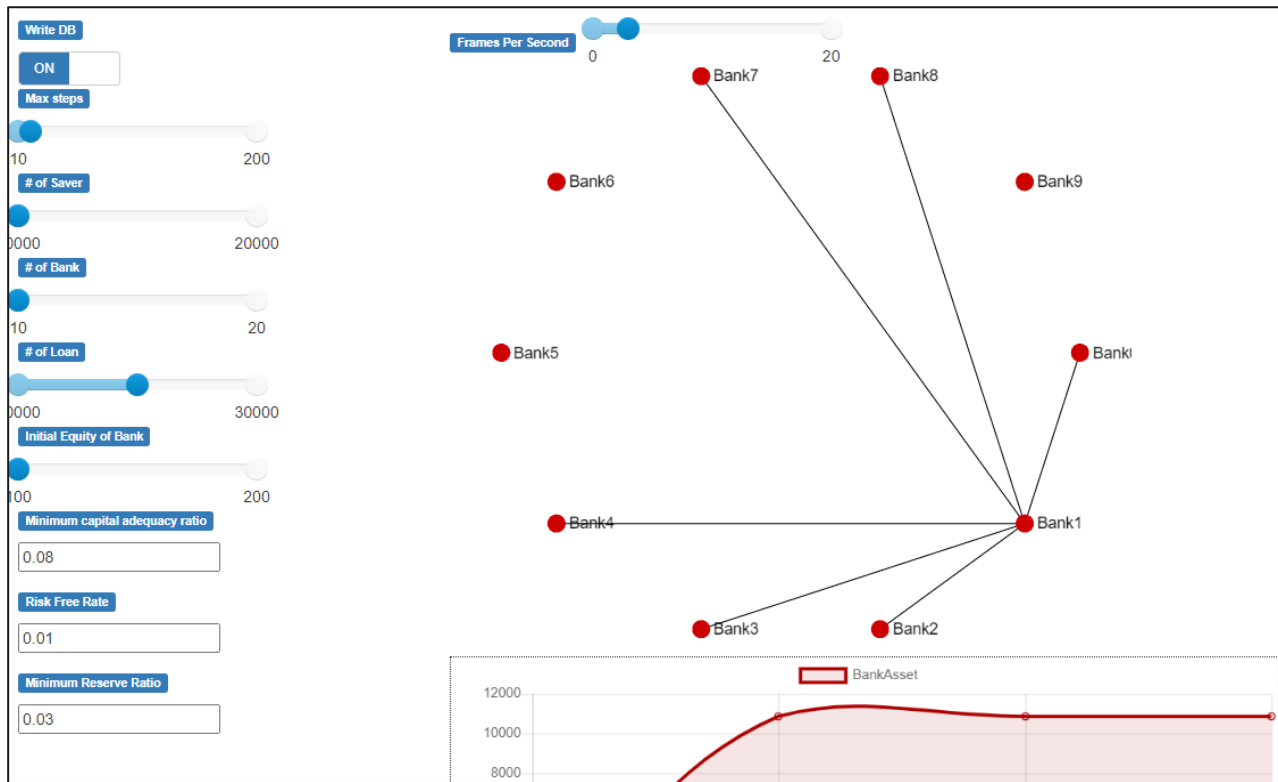
✖

```
(env) C:\Users\byeun\workspace\mesa\notebook>python case6
_onebank_multisavers_multiloans_moreflex.py
```


은행시스템 시뮬레이션

Chan-Lau, Jorge Antonio, ABBA: An Agent-Based Model of the Banking System (September 1, 2015). Available at SSRN:

<https://ssrn.com/abstract=2784228> or
<http://dx.doi.org/10.2139/ssrn.2784228>



ABBA – 행위자 설계

예금 (Saver)
10,000 예금자, 균등 분포
예금:은행 = n:1
연 1% 이자 – 누적되지 않음
매분기 은행 교체확률: uniform[0.0 - 0.2]

은행 (Bank)
10개 은행
초기 자본: 100
MRR 최소 1.5 배 선호
CAR 최대 1.5배 선호
초과 자본은 배당 (충분한 예금액 보유시)
유동성 문제시, 구제금융 신청
지급불능시, 대출회수

대출 (Loan)
20,000 대출, 균등 분포
$n_{\text{대출}} > n_{\text{예금}} \rightarrow$ 은행이 언제나 대출 가능하게끔
부도확률: uniform[0.0- 0.1]
부도 시 대출회수율: 0.4
헐값매각시 보상가격: uniform[0.0-0.1]
위험가중치: 0.5 x 부도확률

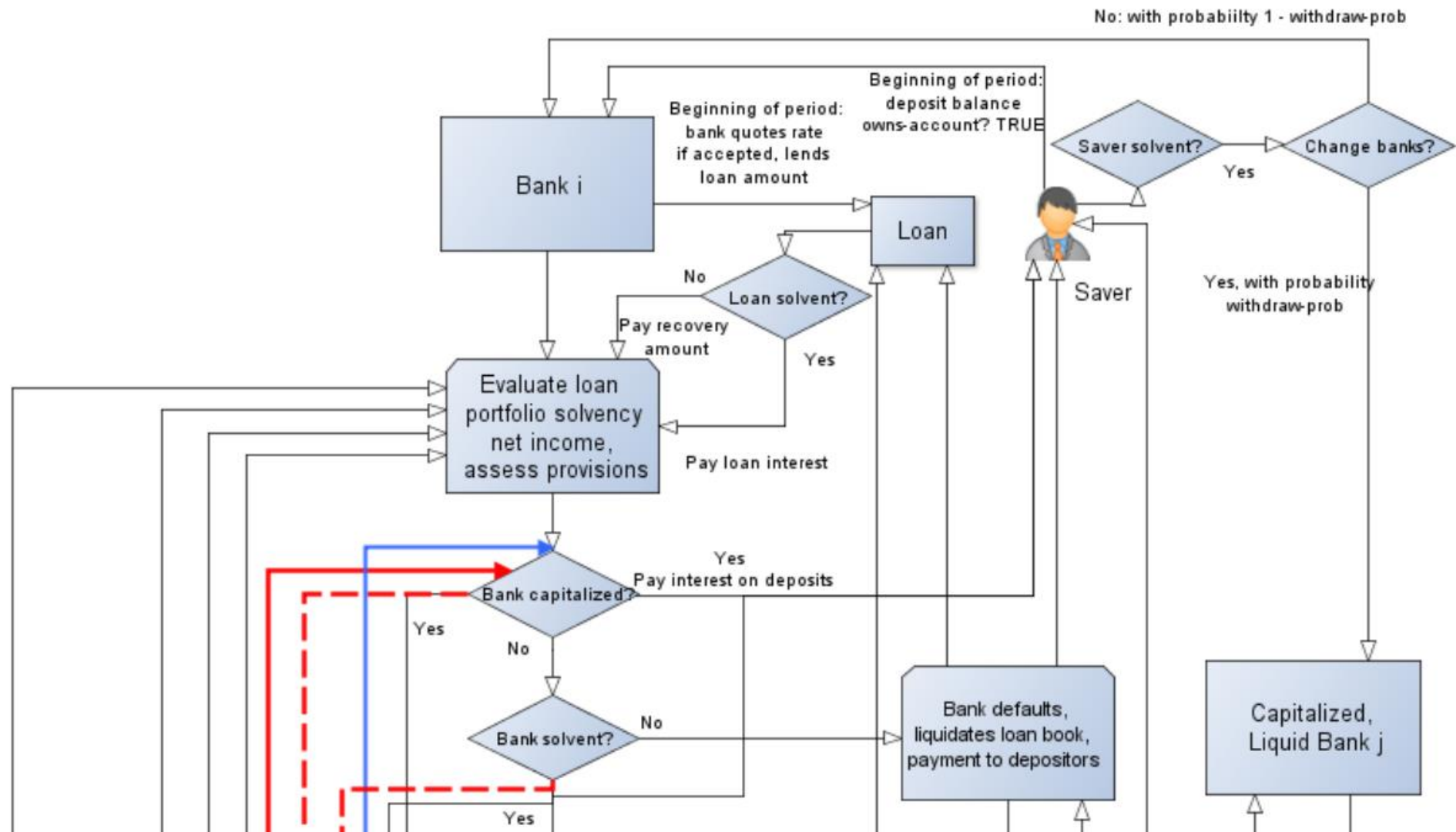
ABBA – 은행시스템 환경

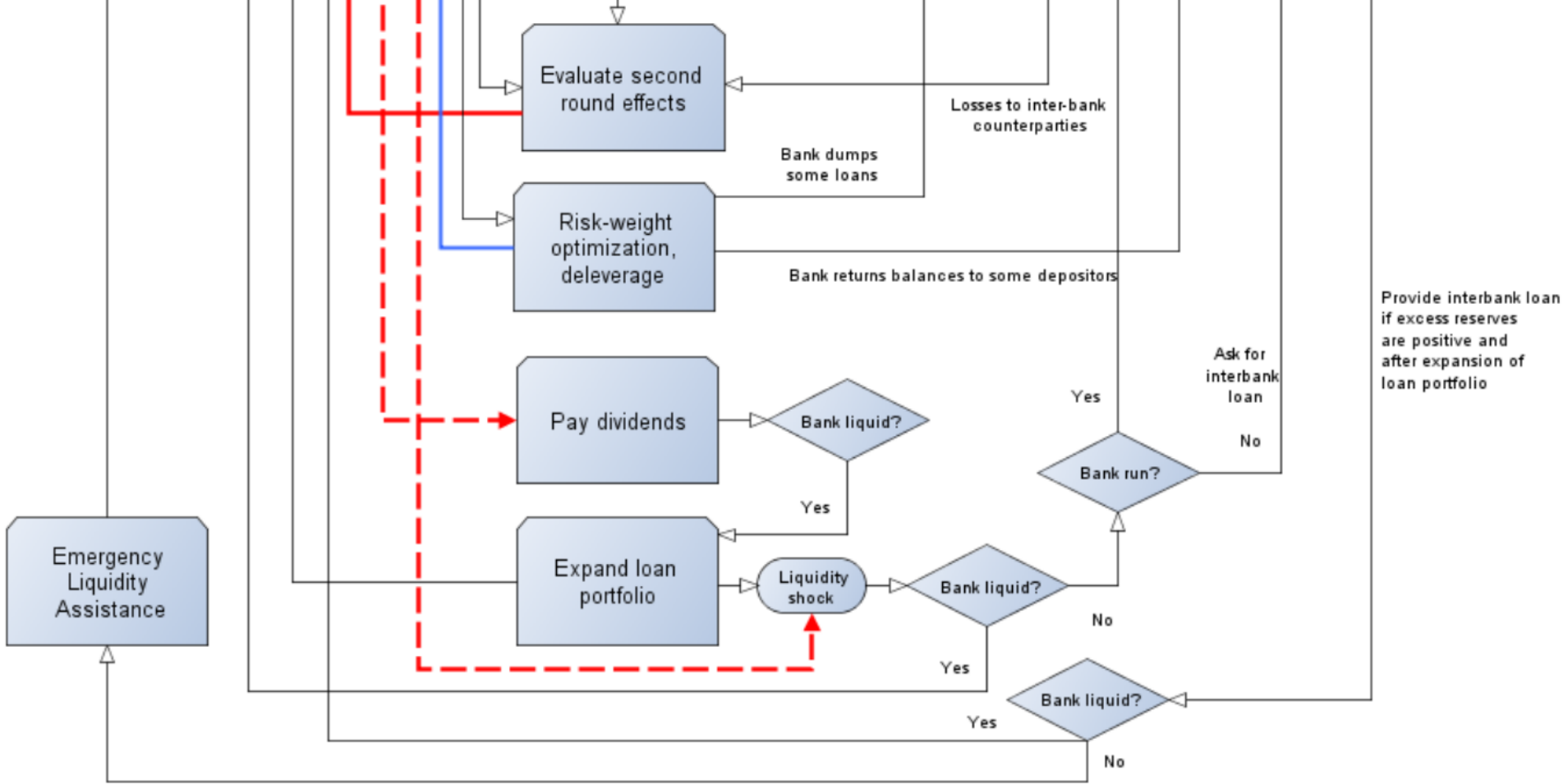
- 은행간 대출 (Interbank Loans)
 - 은행간 대출 이자율 1%
 - 초과 유보금을 보유한 은행은 유동성 문제가 없고 충분한 자본을 보유한 은행에게 대출
- 구제 금융 (Emergency Liquidity Assistance)
 - 자본잠식이나 유동성 문제가 없으면 구제 금융 신청 가능 - 이자율 0%
 - 지급 가능시 다음기에 대출 상환, 대출 연장시 규제 없음
- 정책 규제 (Regulatory Requirements)
 - 자본 준비금 (CAR:자본 대비 위험 가중 자산 비율)에 대한 시나리오 - [0.04, 0.08, 0.12, 0.16]
 - 지급 준비율 (MRR)에 대한 시나리오 - [0.03, 0.045, 0.06]

ABBA – 시뮬레이션

- Step: 240 기 (한 기를 1주로 가정)
- 자기자본비율(CAR) 4 가지 조건 * 지급준비율 (MRR) 3가지 조건 = 12 경우의 수
- 한 경우의 수당 100 번씩 시뮬레이션 반복하여 평균값 (← 확률 모형)
- 처리순서

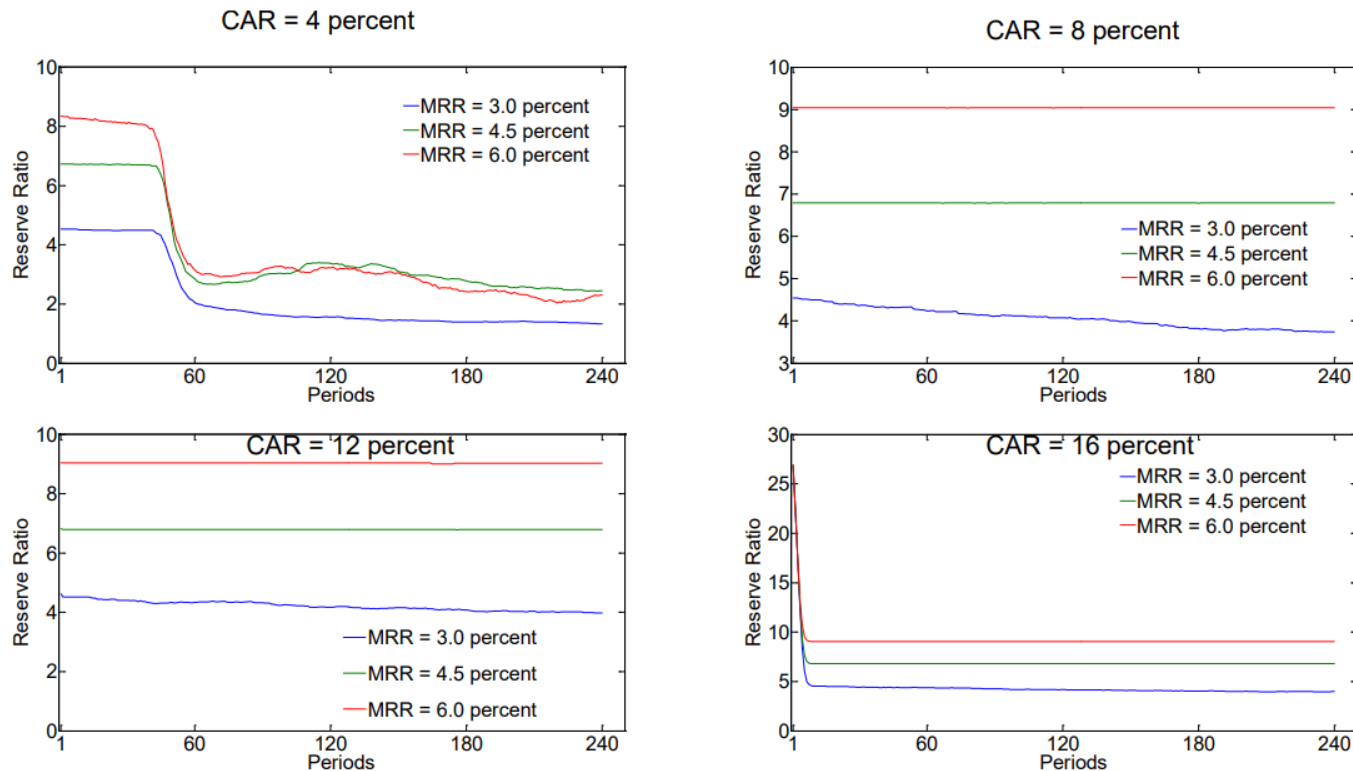
- | | |
|---|---|
| 1 | 행위자 생성, 예금자 할당, 대출 생성 |
| 2 | 대출 손실 계산 – 지급불능시 2차 파급효과 계산 (은행간 대출 손실 계산) |
| 3 | 배당 지급 (충분한 자본을 보유한 은행) |
| 4 | 자본 잠식 시, 위험-가중치 최적화를 통한 자기자본 비율 회복 시도. 지급 불능 시 은행 파산 절차 |
| 5 | 예금 의 은행 이동 후, 은행의 유동성 충격 계산 |
| 6 | 충분한 자본을 보유하고 유동성 문제가 있는 은행은 은행간 대출 신청, 지불가능 하고 자본 잠식 은행은 구제금융 신청, 은행 운영에 실패한 은행은 파산절차 |



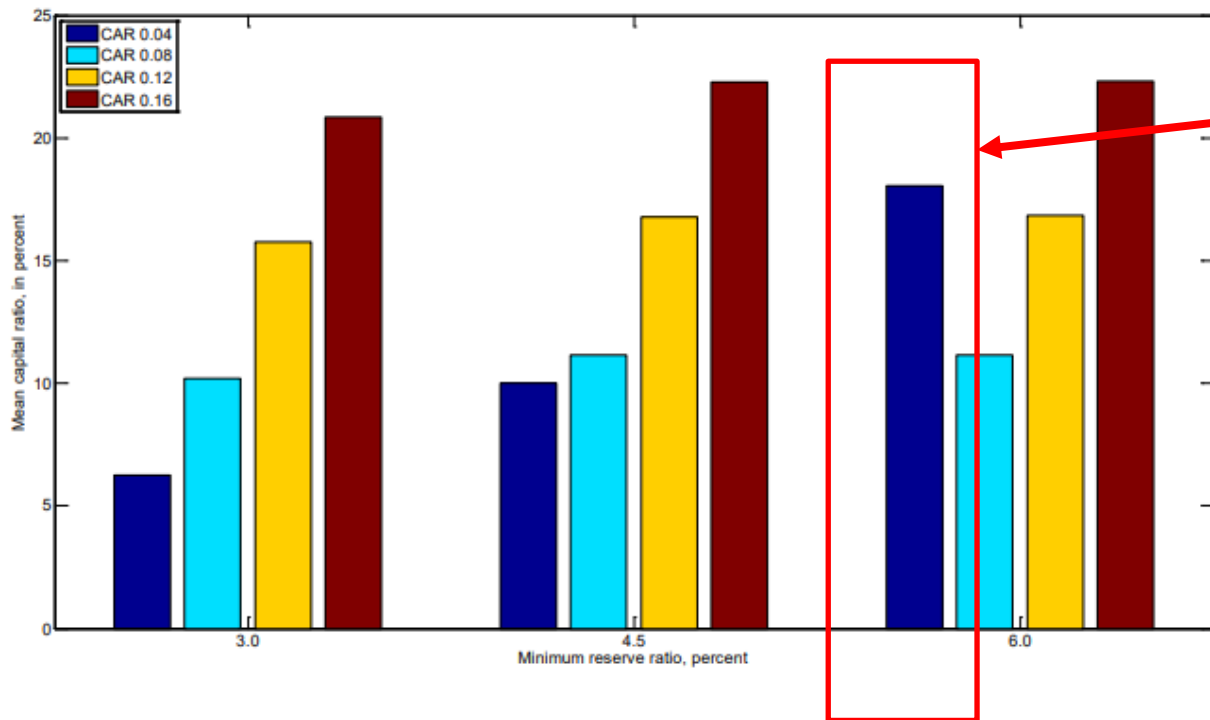


ABBA – 결과 차트: 자기자본비율 규제에 따른 평균 지급준비율 변화

Figure 4. Average reserve ratio for different regulatory requirements, in percent

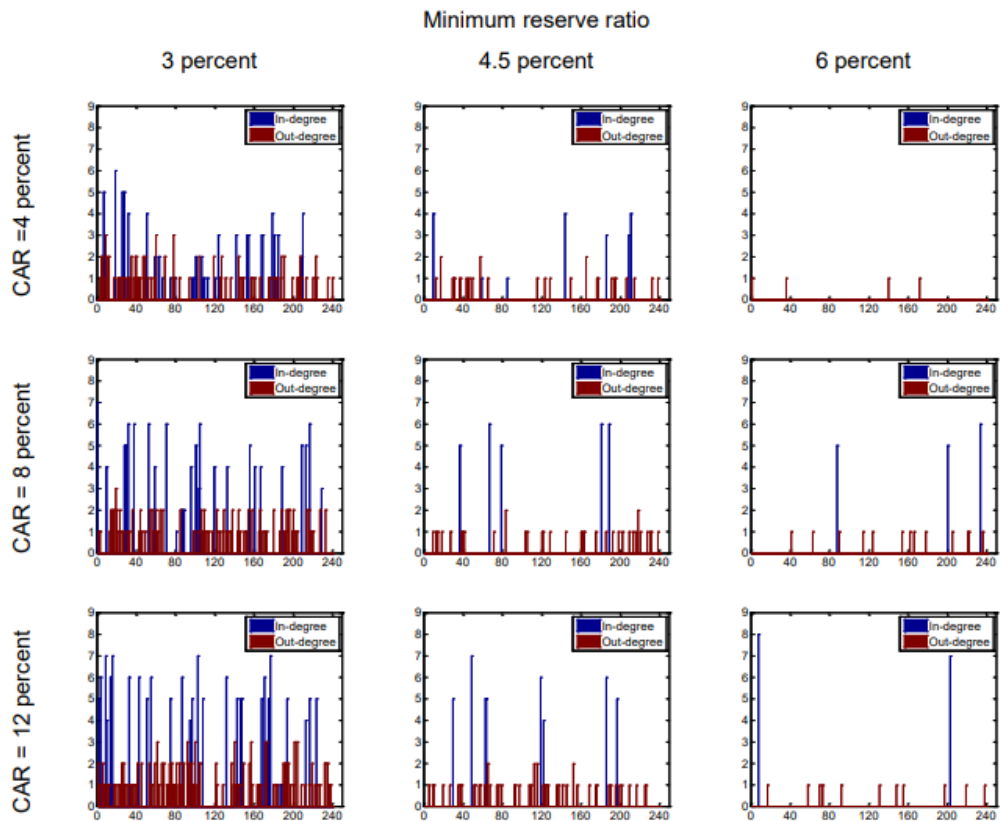


ABBA – 결과 차트: 지급준비율 규제에 따른 평균 자기자본비율 변화



규제 당국이 요구하는 자기자본비율이
낮음에도 높아진 최소 지급준비율
때문에 이익을 최적화 하기 어려움

ABBA – 결과차트: CAR, MRR 변화에 따른 은행간 대출에 대한 빈도 측정



BankSim 소스코드 - 행위자 설계

```
for i in range(self.initial_bank):
    bank = Bank(
        {
            "unique_id": self.next_id(),
            "model": self,
            "equity": 100,
            "rfree": self.rfree,
            "car": self.car,
            "buffer_reserves_ratio": 1.5,
        }
    )
    self.grid.place_agent(bank, i)
    self.schedule.add(bank)

for i in range(self.initial_saver):
    saver = Saver(
        {
            "unique_id": self.next_id(),
            "model": self,
            "balance": 1,
            "owns_account": False,
            "saver_solvent": True,
            "saver_exit": False,
            "withdraw_upperbound": 0.2,
            "exitprob_upperbound": 0.06,
```

```
for i in range(self.initial_loan):
    loan = Loan(
        {
            "unique_id": self.next_id(),
            "model": self,
            "rfree": self.rfree,
            "amount": 1,
            "loan_solvent": True,
            "loan_approved": False,
            "loan_dumped": False,
            "loan_liquidated": False,
            "pdf_upper": 0.1,
            "rcvry_rate": 0.4,
            "firesale_upper": 0.1,
        }
    )
```

BankSim 소스코드 - 시뮬레이션 초기화

```
def initialize_deposit_base(schedule):
    for bank in [x for x in schedule.agents if isinstance(x, Bank)]:
        savers = [x for x in schedule.agents if isinstance(x, Saver) and x.pos == bank.pos]
        for saver in savers:
            saver.bank_id = bank.pos
            saver.owns_account = True
        bank.bank_deposits = sum([x.balance for x in savers])
        bank.bank_reserves = bank.bank_deposits + bank.equity

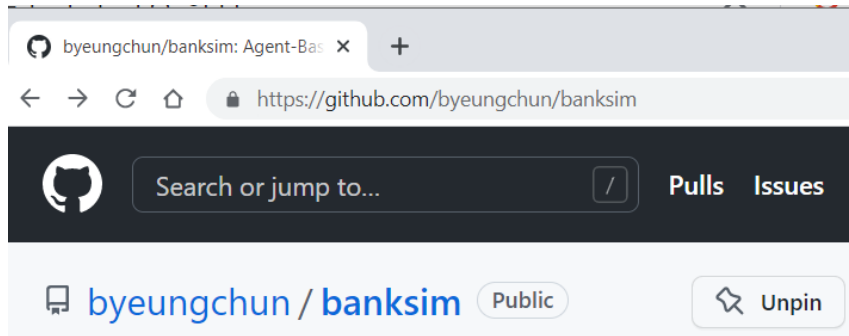
def initialize_loan_book(schedule, car, min_reserves_ratio):
    for bank in [x for x in schedule.agents if isinstance(x, Bank)]:
        bank.bank_reserves = bank.equity + bank.bank_deposits
        bank.calculate_reserve_ratio()
        bank.max_rwa = bank.equity / (1.1 * car)
        interim_reserves = bank.bank_reserves
        interim_deposits = bank.bank_deposits
        interim_reserves_ratio = bank.reserves_ratio
        rwa = 0
        unit_loan = 0
        available_loans = True
```

BankSim 소스코드 - 대출 신용위험 계산

```
bankingsystem
├── __pycache__
├── f1_init_market.py
├── f2_eval_solvency.py
├── f3_second_round_effect.py
├── f4_optimize_risk_weight.py
├── f5_pay_dividends.py
├── f6_expand_loan_book.py
└── f7_eval_liquidity.py

10
11 def calculate_credit_loss_loan_book(schedule, solvent_bank):
12     loans_with_bank = [x for x in schedule.agents if isinstance(x, Loan)
13                        | x.loan_approved and x.loan_solvent]
14     for loan in loans_with_bank:
15         if loan.pdef > random.random():
16             loan.loan_solvent = False
17             # TO DO: change color to magenta
18             loan.rwamount = loan.rweight * loan.amount
19     loans_with_bank_default = [x for x in loans_with_bank if not x.loan_s
20                               # notice that deposits do not change when loans are defaulting
```

소스 저장소: <https://github.com/byeungchun/banksim>



시뮬레이션 개발

- 행위자 정의, 행위 설계
- 행위자간 네트워크
- 반복을 통한 단순 모형 → 복잡 시스템으로 구현
- 개발 툴
 - 시뮬레이션 라이브러리
 - 비주얼라이제이션

정리

- 프로그래밍을 통한 시뮬레이션 실험환경 구축
- 금융시스템에서 발생하는 복잡현상에 대한 재현
- 시뮬레이션으로 실제 현상을 모사하기 위한 핵심 요소
 - 행위자 속성 정의
 - 파라미터 calibration

Build your own matrix! Good Luck



참고자료

- 사회적 원자, 마크 뷰캐넌
- 시스템 다이내믹스 모델링과 시뮬레이션, 곽상만·유재국
- 미시동기와 거시행동, 토마스 셸링
- Agent-Based Computational Economics, <http://www2.econ.iastate.edu/tesfatsi/ace.htm>
- Quantitative Economics with Python, <https://python.quantecon.org/intro.html>
- NetLogo, <https://ccl.northwestern.edu/netlogo/>
- NetLogo 기반 은행시스템 시뮬레이터, Jorge A. Chan-Lau, <https://github.com/jchanlauimf/ABBA>
- Python 기반 은행시스템 시뮬레이터, 권병천, <https://github.com/byeungchun/banksim>