Appendix to

Playing Is Believing: Teaching How Electoral Systems Change Political Outcomes Using a Role-Playing Simulation Game

December 30, 2017

A The Sheets Used in the Game

A.1 Distribution Sheet

Table A1: Distribution Sheet (translated)

Player ID:

Sector	Population	Distributed Resource
Labor union (Sector 1)	40,000	
PTA (Sector 2)	30,000	
Chamber of commerce (Sector 3)	20,000	
Japan Agricultural Co-operatives (Sector 4)	10,000	
Total	100,000	10

Note 1: Magnitude is 3.

Note 2: If tied, winner is randomly decided.

Table A2: Distribution Sheet (in Japanese)

プレイヤー1

団体	人数	配分額
労組	40,000	
PTA	30,000	
商工会議所	20,000	
農協	10,000	
合計	100,000	10

注 *1:* 選挙区定数は3名です。

注 *2:* 同点の場合、当選者は無作為に決まります。

A.2 Result Sheet

Table A3: Result Sheet

		Distribution				Votes					
Player	Cum. W/L	Sec.1	Sec.2	Sec.3	Sec.4	Sec.1	Sec.2	Sec.3	Sec.4	Total	W/L
Cand.1	0	6(5.7)	4(3.8)	0(0.0)	0(0.0)	13333	17077	0	0	30410	Lose
Cand.2	2	4(5.7)	4(3.8)	2(1.9)	1(0.8)	8655	12923	20000	3077	44655	Win
Cand.3	1	8(7.7)	0(0.0)	0(0.0)	2(1.8)	18012	0	0	6923	24935	Lose
Competition Cost		0.3	0.2	0.1	0.2						

Table A4: Result Sheet (in Japanese)

		配分額				得票数					
候補者	当選回数	労組	PTA	商工会議所	農協	労組	PTA	商工会議所	農協	合計	結果
候補者 1	0	6(5.7)	4(3.8)	0(0.0)	0(0.0)	13333	17077	0	0	30410	落選
候補者 2	2	4(5.7)	4(3.8)	2(1.9)	1(0.8)	8655	12923	20000	3077	44655	当選
候補者 3	1	8(7.7)	0(0.0)	0(0.0)	2(1.8)	18012	0	0	6923	24935	落選
選挙費用		0.3	0.2	0.1	0.2						

B A Numerical Proof of Expected Result

Let d_i and c denote the distribution vectors and the competiton cost, respectively.

$$\begin{aligned}
d_1 &= (4, 3, 2, 1) \\
d_2 &= (4, 3, 2, 1) \\
d_3 &= (x_1, x_2, x_3, x_4) \\
10 &= \sum d_1 = \sum d_2 = \sum d_3 \\
c &= 0.1
\end{aligned} \tag{1}$$

With d_1 and d_2 fixed, the number of possible allocations for d_3 is 286. We calculated the number of votes won by Candidate 3 with the following R code.

```
test.df <- expand.grid(0:10, 0:10, 0:10, 0:10)
colnames(test.df) <- c("Sec1", "Sec2", "Sec3", "Sec4")
test.df <- test.df[rowSums(test.df) == 10, ]

calc.func <- function(vec, cost = .1) {
    pop <- c(40000, 30000, 20000, 10000)
    result <- rep(NA, 4)
    temp.df <- data.frame("Sec1" = c(4, 4, vec[1]),
                          "Sec2" = c(3, 3, vec[2]),
                          "Sec3" = c(2, 2, vec[3]),
                          "Sec4" = c(1, 1, vec[4]))
    cost.vec <- rep(NA, 4)
    for (i in 1:4) {
        competition <- sum(temp.df[, i] != 0)
        cost.vec[i] <- sum(temp.df[, i] != 0)
    }
    cost.vec <- cost.vec * cost
    for (i in 1:4) {
        temp.df[, i] <- temp.df[, i] - cost.vec[i]
        temp.df[temp.df[, i] < 0, i] <- 0
        result[i] <- pop[i] *
                     (temp.df[3, i]/sum(temp.df[, i]))
    }
    return(sum(result))
}

result.vec <- rep(NA, nrow(test.df))
for (i in seq_along(result.vec)) {
    result.vec[i] <- calc.func(unlist(test.df[i, ]), cost = .1)
}

print(test.df[max(result.vec) == result.vec, ])
```

This simulation confirmed that the distribution vector of Candidate 3 that maximizes his votes is $d_3 = (4, 3, 2, 1)$.[1]

C Source Code

```
Game.init <- function(nSec = 4, nCand = 5, M = 1,
                      sec.name = c("Sector1", "Sector2",
                                   "Sector3", "Sector4"),
                      sec.dist = c(400000, 300000, 200000, 100000),
                      Players  = c("Cand.1", "Cand.2", "Cand.3",
                                   "Cand.4", "Cand.5")
){

        result.list <- list()
        result.list$round    <- 0
        result.list$nSec     <- nSec
        result.list$nCand    <- nCand
        result.list$M        <- M
        result.list$secName  <- sec.name
        result.list$secDist  <- sec.dist
        result.list$player   <- Players
        result.list$n.win    <- rep(0, nCand)

        return(result.list)
}

create.empty.df <- function(n.sec, n.cand, sec.name, player) {
        body <- rep(0, n.sec)
        for (i in 1:(n.cand - 1)) {
```

[1] If parameter c were different, maximizing vector would be different. If $c > \frac{5}{27}$, the maximizing vector is $d_3 = (3, 3, 2, 2)$.

```
                body <- rbind(body, rep(0, n.sec))
        }

        result.df <- cbind(player,
                           body,
                           as.numeric(rep(0, n.cand)),
                           as.numeric(rep(0, n.cand)))

        result.df <- data.frame(result.df)
        result.df[, n.sec + 2] <- as.numeric(result.df[, n.sec + 2])
        result.df[, n.sec + 3] <- as.numeric(result.df[, n.sec + 3])
        colnames(result.df) <- c("ID", sec.name, "Vote", "W/L")

        return(result.df)
}

do.election <- function(distribution, out = FALSE) {

    GC[[1]]  <<- GC[[1]] + 1

    round    <- GC[[1]]
    n.sec    <- GC[[2]]
    n.cand   <- GC[[3]]
    M        <- GC[[4]]
    sec.name <- GC[[5]]
    sec.dist <- GC[[6]]
    player   <- GC$player

    if (out) {
        dist <- read.csv(distribution, header = TRUE)
        dist <- dist[, -1]
    } else {
        dist <- data.frame(distribution)
    }

    colnames(dist) <- sec.name

    comp.cost <- dist != 0
    comp.cost <- colSums(comp.cost) * 0.1

    for (i in 1:n.cand) {
        dist[i, ] <- dist[i, ] - comp.cost
    }
    dist[dist < 0] <- 0
    result.df <- create.empty.df(n.sec, n.cand,
                                 sec.name, player)

    for (i in 1:n.sec) {
        prob <- as.vector(dist[, i])
        get.vote <- round(sec.dist[i] * (prob / sum(prob)), 0)
        result.df[, (i + 1)] <- get.vote
    }

    for (i in 1:n.cand) {
        get.vote <- sum(result.df[i, 2:(n.sec + 1)])
        result.df[i, (n.sec + 2)] <- get.vote
    }

    temp.win <- rank(result.df[, (n.sec + 2)],
                     ties.method = "random")
    win.lose <- c()

    for (j in seq_along(temp.win)) {
        if (temp.win[j] > (n.cand - M)) {
            win.lose[j] <- "Win"
            GC$n.win[j] <<- GC$n.win[j] + 1
        } else {
            win.lose[j] <- "Lose"
        }
    }
    result.df[, (n.sec + 3)] <- factor(win.lose)
    result.df <- cbind(result.df[, 1],
                       GC$n.win,
                       dist,
                       result.df[, -1])
    colnames(result.df)[1:2] <- c("Player", "# of Win")
    print(paste("Round:", round))
    return(result.df)
}

# MMD-SNTV
GC <- Game.init(nSec = 4, nCand = 5, M = 1,
                sec.name = c("Sector 1", "Sector 2",
                             "Sector 3", "Sector 4"),
                sec.dist = c(400000, 300000, 200000, 100000),
                Players  = c("Cand.1", "Cand.2", "Cand.3",
                             "Cand.4", "Cand.5"))

dist.mat <- matrix(c(8, 0, 0, 2,
                     0, 6, 4, 0,
                     4, 3, 2, 1,
                     3, 5, 1, 1,
                     3, 3, 2, 2),
                   nrow = 5, byrow = TRUE)

do.election(distribution = dist.mat) # Round 1


# SMD
GC <- Game.init(nSec = 4, nCand = 3, M = 1,
                sec.name = c("Sector 1", "Sector 2",
                             "Sector 3", "Sector 4"),
                sec.dist = c(400000, 300000, 200000, 100000),
                Players  = c("Cand.1", "Cand.2", "Cand.3"))

dist.mat <- matrix(c(4, 3, 2, 1,
                     8, 2, 0, 0,
                     3, 4, 1, 2),
                   nrow = 3, byrow = TRUE)

do.election(distribution = dist.mat) # Round 1
```