Gender biases in dialogue transitions

Introduction

As well as the amount of dialogue given to female characters, studies of film and television have shown biases in who they speak to. The "Bechdel test" or "Bechdel-Wallace test" is a popular illustration of a bias against females talking to other females. The test originated as an idea in a comic strip (Bechdel, 1986), and has become more widely known. A film passes the Bechdel test if it depicts:

1. At least two named female characters ...
2. ... who talk to each other
3. ... about something other than a man.

For example, we are reasonably sure that *The Secret of Monkey Island* does not pass the Bechdel test, simply because there are so few female characters and there are no scenes with two female characters in. According to our corpus, there apparently is only one case of a transition between two females in any of the three Monkey Island games, though this turns out to be the following from Monkey Island 2:

> "Woman 1 watching spit contest": "(claps)"
>
> "Woman 2 watching spit contest": "(claps)"

There is no linguistic content, and these women are not named. So all three games apparently fail the Bechdel test.

The Bechdel test has been influential and revealing, but there are issues with applying it as a measure of gender bias in video game dialogue. The first issue is that the Bechdel test is very strict. The data in our corpus is often a sample, and sometimes only one possible play-through. Other games have procedural elements. For any given game, it would be very hard to prove that there is *no possibility* of dialogue between two women.

The second issue is that video games often have more dialogue than the average TV show or film. So the probability of there being *no* female-female dialogue is very low. So most games would pass the test. However, this doesn't mean that there are no problems, and it doesn't show which games have more problems than others.

The third issue is that it's possible for a game to potentially pass the Bechdel test, while not actually passing it for a specific player's experience. The test is well suited to short scripts with canonical forms (like TV, Film, books), but conceptually more difficult to apply to interactive texts like video games.

There are alternative measures, such as the "Mako Mori test" which test whether a female character has a narrative ark. However, this also has the issues above.

Therefore, for this study it makes more sense to apply a probabilistic test that shows a bias away from an expected value. This would help quantify the relative degree of bias between games. So, in the following sections, we estimate the frequencies of dialogue transitions between characters of different genders and assess these in relation to baselines calculated using a permutation procedure.

Measuring transitions

Data for most games in the corpus captures the local order of dialogue. That is, while players might experience scenes in a different order (e.g. by choosing to speak to characters in a different order), the order within a scene is consistent. For these games, we want to count the number of dialogue transitions between characters of different genders. This can be done by iterating through the game script and counting transitions from one character to the next. This is similar to previous "consecutive" approaches to automating the Betchdel test that build an interaction network between characters based on a consecutive lines of dialogue (Agarwal et al., 2015).

However, neither games nor films have a single continuous conversation. Transitions should only be counted for characters who are actually talking *to each other*. Automated approaches applied to film assume that all characters within a scene are effectively talking to each other, so count only the transitions that happen within scenes (Weng et al., 2009; Agarwal et al., 2015). Some games have scene or location transitions, encoded in "LOCATION" lines. During parsing, games where conversation boundaries were detectable but not labelled were indicated with an ACTION line with the value "—". However, scenes within films are often much shorter than in games (there may be long periods in a game without a narrative "cut"). Furthermore, dialogue is often in short bursts, separated by longer game playing sections. Therefore, in addition to scenes and locations, we treat action descriptions as scene boundaries.

A python script (`processing/DialogueTransitions.py`) stepped through the script, identifying dialogue transitions between two non-identical characters. It looked up these characters' relative gender categories, and incremented a count for transitions from the first to the second category. In dialogue trees with choices, the algorithm counted all possible dialogue transitions that a player might experience (considering only branching options without re-visiting earlier parts in the tree). The script output the frequency of transitions between each gender category (stored as `transitions.csv`) and the sequence of individual speaker categories and scene breaks (stored as `transitions_all.csv`) for use in the premutation test (see below).

The automated method here will obviously not always be accurate. Agarwal et al., report that their method of scene boundary detection has accuracies of around 65% for identifying whether women talk to each other, so it is likely that there will be many errors. However, the aim of the current study is to identify large-scale gender biases in lingusitic behaviour. So we proceed with the assumption that the errors will be unbiased by gender. That is, the scene boundary cues we use are not distributed differently depending on the gender of the surrounding characters.

Calculating expected transitions

For each game, we now have the empirical number of transitions between each gender category. For example, Final Fantasy VII has the following transition frequencies, where the rows indicate the gender of the character before the transition and the columns indicate the gender of the character after the transition:

```
##        male female
## male   2666   1150
## female 1199    225
```

The frequency of female-to-female dialogue is low, but is it significantly lower than we would expect by chance? This is more difficult to answer than it might appear. A game that has no biases in transitions between female characters won't necessarily have an equal number of each type of transition (female-to-female, female-to-male, etc.). For example, imagine a game with many more male characters than female characters. It is likely that this would have more male-to-male transitions than female-to-female transitions, even if the creators were designing many opportunities for female characters to talk to each other. So while a low proportion of female-to-female transitions might be evidence for a general bias against female dialogue, it is not necessarily evidence for a bias against *transitions between* female characters. This requires comparing the transitions we observe to a "baseline" of values we would expect by chance if there was no bias in who talks to who.

The calculation of this baseline is difficult to compute analytically, since it depends on the number of male and female lines, and the number of scene boundaries. Instead, we use permutation: the full sequence of gender categories and scene boundaries for a game is randomly re-shuffled. This creates a hypothetical script with the same numbers of lines for each gender category and the same number of scene boundaries. This step is repeated many times to produce many scripts (10,000 times for the analysis of all games together, and 1,000 times for each individual game analysis). On average, we would expect these hypothetical scripts to break any implicit gender biases in dialogue transitions (they should have an 'unbiased' frequency of transitions between each gender category). The frequency of transitions are be re-calculated for each hypothetical script to get a distribution of permuted transition frequencies. These can be compared to the "real" frequencies. From this we can calculate the following for a given transition between gender categories:

- The mean frequency of permuted transitions, which serve as the expected values if there was no

gender bias.

- A z-score which represents a normalised distance between the "real" value and the expected values.
- A p-value: The p-value represents the proportion of hypothetical scripts that exhibited a more extreme frequency than the empirical frequency. This serves as an indicator of the likelihood that the empirical frequency would be produced by chance.

There are four possible transition types between two genders, but the frequencies for these are obviously not independent of each other. Therefore, we focus on two measures: the proportion of female-to-female transitions (to reflect the Bechdel test), and the proportion of male-to-male transitions (as the orthogonal measure). In principle, these measures are relatively independent, so a game could have a high or low value for either measure.

Load libraries

```
library(rjson)
library(Gmisc)
library(dplyr)
library(RColorBrewer)
library(grid)
library(ggplot2)
library(ggrepel)
```

Set some parameters for the permutation tests:

```
numberOfPermutationsForAllGames = 10000
numberOfPermutationsForEachGame = 1000
```

Load data

```
folders = list.dirs("../data", recursive = T)
folders = folders[sapply(folders,function(X){
  "transitions.csv" %in% list.files(X)
})]


allGames= NULL
transitionStrings = NULL
for(folder in folders){
  print(folder)
  shortName = tail(strsplit(folder,"/")[[1]],1)
  js = fromJSON(file = paste0(folder,"/meta.json"))
  alternativeMeasure = FALSE
  if(!is.null(js$alternativeMeasure)){
    alternativeMeasure = js$alternativeMeasure
  }
  suitableForTransitions = TRUE
  if(!is.null(js$sourceFeatures$dialogueOrder)){
      suitableForTransitions = js$sourceFeatures$dialogueOrder
  }

  if(suitableForTransitions & (!alternativeMeasure)){
    sbcFile = paste0(folder,"/transitions.csv")
    if(file.exists(sbcFile)){
      d = read.csv(sbcFile,stringsAsFactors = F)
      if(nrow(d)>0){
        d$shortName = shortName
      }
      allGames = rbind(allGames,d)
    }
    tsFile = paste0(folder,"/transitions_all.txt")
    if(file.exists(tsFile)){
      ts = suppressWarnings(readLines(tsFile)[1])
      transitionStrings = rbind(transitionStrings,
                    data.frame(folder=folder,series=d$series[1],
                               game=d$game[1],
                               ts = ts,
                               stringsAsFactors = F))
    }
  }
```

```
}
```

```
## [1] "../data/ChronoTrigger/ChronoTrigger"
## [1] "../data/DragonAge/DragonAge2"
## [1] "../data/DragonAge/DragonAgeInquisition"
## [1] "../data/DragonAge/DragonAgeOrigins_B"
## [1] "../data/ElderScrolls/Daggerfall"
## [1] "../data/ElderScrolls/Morrowind"
## [1] "../data/ElderScrolls/Oblivion"
## [1] "../data/ElderScrolls/Skyrim"
## [1] "../data/FinalFantasy/FFI"
## [1] "../data/FinalFantasy/FFII"
## [1] "../data/FinalFantasy/FFIII"
## [1] "../data/FinalFantasy/FFIV_DS"
## [1] "../data/FinalFantasy/FFIX"
## [1] "../data/FinalFantasy/FFIX_B"
## [1] "../data/FinalFantasy/FFV"
## [1] "../data/FinalFantasy/FFVI"
## [1] "../data/FinalFantasy/FFVII"
## [1] "../data/FinalFantasy/FFVII_Remake"
## [1] "../data/FinalFantasy/FFVIII"
## [1] "../data/FinalFantasy/FFX"
## [1] "../data/FinalFantasy/FFX_B"
## [1] "../data/FinalFantasy/FFX2"
## [1] "../data/FinalFantasy/FFXII"
## [1] "../data/FinalFantasy/FFXII_B"
## [1] "../data/FinalFantasy/FFXIII"
## [1] "../data/FinalFantasy/FFXIII-2"
## [1] "../data/FinalFantasy/FFXIII-LR"
## [1] "../data/FinalFantasy/FFXIV"
## [1] "../data/FinalFantasy/FFXV"
## [1] "../data/Horizon/HorizonZeroDawn"
## [1] "../data/KingdomHearts/KingdomHearts"
## [1] "../data/KingdomHearts/KingdomHearts_B"
## [1] "../data/KingdomHearts/KingdomHearts2"
## [1] "../data/KingdomHearts/KingdomHearts3"
## [1] "../data/KingdomHearts/KingdomHearts3D"
## [1] "../data/KingsQuest/KingsQuest1"
## [1] "../data/KingsQuest/KingsQuest2"
## [1] "../data/KingsQuest/KingsQuest3"
## [1] "../data/KingsQuest/KingsQuest4"
## [1] "../data/KingsQuest/KingsQuest5"
## [1] "../data/KingsQuest/KingsQuest6"
## [1] "../data/KingsQuest/KingsQuest7"
## [1] "../data/KingsQuest/KingsQuest8"
## [1] "../data/KingsQuest/KingsQuestChapters"
## [1] "../data/MassEffect/MassEffect1"
## [1] "../data/MassEffect/MassEffect1B"
## [1] "../data/MassEffect/MassEffect2"
## [1] "../data/MassEffect/MassEffect3"
## [1] "../data/MassEffect/MassEffect3C"
## [1] "../data/MassEffect/MassEffectAndromeda"
## [1] "../data/MonkeyIsland/MonkeyIsland2"
## [1] "../data/MonkeyIsland/TheCurseOfMonkeyIsland"
## [1] "../data/MonkeyIsland/TheSecretOfMonkeyIsland"
## [1] "../data/Persona/Persona3"
## [1] "../data/Persona/Persona4"
## [1] "../data/Persona/Persona5"
```

```
## [1] "../data/Persona/Persona5B"
## [1] "../data/StardewValley/StardewValley"
## [1] "../data/StarWarsKOTOR/StarWarsKOTOR"
## [1] "../data/SuperMarioRPG/SuperMarioRPG"
```

```
# Trim final stroke to make consistent
transitionStrings$folder = gsub("/$","",transitionStrings$folder)
allGames$folder = gsub("/$","",allGames$folder)
```

Analogue of the Betchdel test: try to find games with no transitions between two female characters:

```
hasFemaleToFemale = sapply(unique(allGames$folder), function(fld){
  any(allGames[allGames$folder==fld,]$from=="female" & allGames[allGames$folder==fld,]$to=="female",
})

t(t(names(hasFemaleToFemale[!hasFemaleToFemale])))
```

```
##      [,1]
## [1,] "../data/FinalFantasy/FFI"
## [2,] "../data/MassEffect/MassEffect1B"
## [3,] "../data/MassEffect/MassEffect2"
## [4,] "../data/MassEffect/MassEffect3C"
## [5,] "../data/MonkeyIsland/TheCurseOfMonkeyIsland"
## [6,] "../data/MonkeyIsland/TheSecretOfMonkeyIsland"
```

Permutation tests

Function to count different types of transition and run the permutation tests:

```
getTransitions = function(allGames, raw=F){

  maleToMale = sum(allGames[allGames$from=="male" & allGames$to=="male",]$frequency)
  maleToFemale = sum(allGames[allGames$from=="male" & allGames$to=="female",]$frequency)
  femaleToMale = sum(allGames[allGames$from=="female" & allGames$to=="male",]$frequency)
  femaleToFemale = sum(allGames[allGames$from=="female" & allGames$to=="female",]$frequency)

  if(raw){
    maleToMaleP = maleToMale
    maleToFemaleP = maleToFemale
    femaleToMaleP = femaleToMale
    femaleToFemaleP = femaleToFemale
  }
  else{
    total = sum(maleToMale,maleToFemale,femaleToMale,femaleToFemale)
    maleToMaleP = maleToMale/(maleToMale+maleToFemale)
    maleToFemaleP = maleToFemale/(maleToMale+maleToFemale)
    femaleToMaleP = femaleToMale/(femaleToMale+femaleToFemale)
    femaleToFemaleP = femaleToFemale/(femaleToMale+femaleToFemale)
  }

  transitionTable =matrix(c(maleToMaleP, femaleToMaleP, maleToFemaleP, femaleToFemaleP),nrow=2)
  rownames(transitionTable) = c("m","f")
  colnames(transitionTable) = c("m","f")

  return(transitionTable)

}
```

Functions to extract and print the stats:

```
permuteTransitionString = function(X){
  # permute order
  X = sample(X)
  # calculate table of transitions using lag
  tab = table(X[1:(length(X)-1)], X[2:length(X)])
  tab = tab[c("m","f"),c("m","f")]
  tab[is.na(tab)] = 0
  transitionProbs = prop.table(tab,1)
}

getZP = function(tpPerm,trueProb){
  Z = (trueProb - mean(tpPerm))/sd(tpPerm)
  sx = sum(tpPerm >trueProb)
  P = 1/length(tpPerm)
  if(sx>0){
    P = sx/length(tpPerm)
  }
  if(Z<0){
    P = 1 - P
  }
  if(P==0){
    P = 1/length(tpPerm)
  }
  return(c(mean = mean(tpPerm), z = Z,p = P))
}

getPermutedStats = function(ts, trueTransitionProbs,numPerm){
  lines = strsplit(ts,"")[[1]]
  transProbsPerm = replicate(numPerm, permuteTransitionString(lines))
  m2m = getZP(transProbsPerm["m","m",],trueTransitionProbs["m","m"])
  m2f = getZP(transProbsPerm["m","f",],trueTransitionProbs["m","f"])
  f2f = getZP(transProbsPerm["f","f",],trueTransitionProbs["f","f"])
  f2m = getZP(transProbsPerm["f","m",],trueTransitionProbs["f","m"])

  return(c(m2m, f2m, m2f, f2f))
}
```

Run transition permutation tests for the corpus as a whole:

```
trueTransitions.AllGames = getTransitions(allGames)
trueTransitions.AllGames.Raw = getTransitions(allGames,raw=TRUE)

trueTransitionsString.AllGames = paste0(transitionStrings$ts)
permutedTransitionStats.AllGames =
  getPermutedStats(trueTransitionsString.AllGames,
                   trueTransitions.AllGames,
                   numberOfPermutationsForAllGames)
names(permutedTransitionStats.AllGames) = paste(
  rep(c("m2m", "f2m", "m2f", "f2f"),each=3),
  names(permutedTransitionStats.AllGames),sep=".")

t(t(permutedTransitionStats.AllGames))
```

```
##                 [,1]
## m2m.mean   0.5678064
## m2m.z     10.3953934
## m2m.p      0.0001000
## f2m.mean   0.5681016
## f2m.z      9.2559500
## f2m.p      0.0001000
## m2f.mean   0.4321936
## m2f.z    -10.3953934
## m2f.p      0.0001000
## f2f.mean   0.4318984
## f2f.z     -9.2559500
## f2f.p      0.0001000
```

Stats for each game:

```
set.seed(238)
print("Running stats ...")
```

```
## [1] "Running stats ..."
```

```
permutationResults = NULL
for(folder in unique(allGames$folder)){
  print(folder)
  dx = allGames[allGames$folder == folder,]
  trueTransitionProbs = getTransitions(dx)
  trueTransitionProbsFlat = as.vector(trueTransitionProbs)
  names(trueTransitionProbsFlat) = c("m2m", "f2m", "m2f", "f2f")

  trueTransitionRaw = getTransitions(dx,raw = T)
  trueTransitionRawFlat = as.vector(trueTransitionRaw)
  names(trueTransitionRawFlat) = c("m2m", "f2m", "m2f", "f2f")

  if(any(is.nan(trueTransitionProbs))){
    print(paste("No data for ",folder))
  } else{
    trueTransitionString = transitionStrings[transitionStrings$folder==folder,]$ts
    permutedTransitionStats = getPermutedStats(trueTransitionString,trueTransitionProbs,numberOfPerm
    names(permutedTransitionStats) = paste(rep(c("m2m", "f2m", "m2f", "f2f"),each=3),names(permutedT

    res = data.frame(folder=dx$folder[1],
                     series = dx$series[1],
                     game = dx$game[1],
                     shortName = dx$shortName[1])
```

```
    tt = matrix(trueTransitionProbsFlat,nrow=1)
    colnames(tt) = names(trueTransitionProbsFlat)
    res = cbind(res,tt)

    ttr = matrix(trueTransitionRawFlat,nrow=1)
    colnames(ttr) = paste0(names(trueTransitionRawFlat),".raw")
    res = cbind(res,ttr)

    pt = matrix(permutedTransitionStats,nrow=1)
    colnames(pt) = names(permutedTransitionStats)
    res = cbind(res,pt)

    permutationResults = rbind(permutationResults,res)
  }

}
```

```
## [1] "../data/ChronoTrigger/ChronoTrigger"
## [1] "../data/DragonAge/DragonAge2"
## [1] "../data/DragonAge/DragonAgeOrigins_B"
## [1] "../data/ElderScrolls/Skyrim"
## [1] "../data/FinalFantasy/FFI"
## [1] "../data/FinalFantasy/FFII"
## [1] "../data/FinalFantasy/FFIV_DS"
## [1] "../data/FinalFantasy/FFIX_B"
## [1] "../data/FinalFantasy/FFV"
## [1] "../data/FinalFantasy/FFVI"
## [1] "../data/FinalFantasy/FFVII"
## [1] "../data/FinalFantasy/FFVII_Remake"
## [1] "../data/FinalFantasy/FFVIII"
## [1] "../data/FinalFantasy/FFX_B"
## [1] "../data/FinalFantasy/FFX2"
## [1] "../data/FinalFantasy/FFXII_B"
## [1] "../data/FinalFantasy/FFXIII"
## [1] "../data/FinalFantasy/FFXIII-2"
## [1] "../data/FinalFantasy/FFXIII-LR"
## [1] "../data/FinalFantasy/FFXIV"
## [1] "../data/FinalFantasy/FFXV"
## [1] "../data/Horizon/HorizonZeroDawn"
## [1] "../data/KingdomHearts/KingdomHearts_B"
## [1] "../data/KingdomHearts/KingdomHearts2"
## [1] "../data/KingdomHearts/KingdomHearts3"
## [1] "../data/KingdomHearts/KingdomHearts3D"
## [1] "../data/KingsQuest/KingsQuest6"
## [1] "../data/KingsQuest/KingsQuest7"
## [1] "../data/MassEffect/MassEffect1B"
## [1] "No data for  ../data/MassEffect/MassEffect1B"
## [1] "../data/MassEffect/MassEffect2"
## [1] "../data/MassEffect/MassEffect3C"
## [1] "No data for  ../data/MassEffect/MassEffect3C"
## [1] "../data/MonkeyIsland/MonkeyIsland2"
## [1] "../data/MonkeyIsland/TheCurseOfMonkeyIsland"
## [1] "../data/MonkeyIsland/TheSecretOfMonkeyIsland"
## [1] "../data/Persona/Persona3"
## [1] "../data/Persona/Persona4"
## [1] "../data/Persona/Persona5B"
## [1] "../data/StardewValley/StardewValley"
## [1] "../data/StarWarsKOTOR/StarWarsKOTOR"
```

```
## [1] "../data/SuperMarioRPG/SuperMarioRPG"
```

```
permutationResults$diffExpEmp.m2m = permutationResults$m2m - permutationResults$m2m.mean
permutationResults$diffExpEmp.f2f = permutationResults$f2f - permutationResults$f2f.mean
```

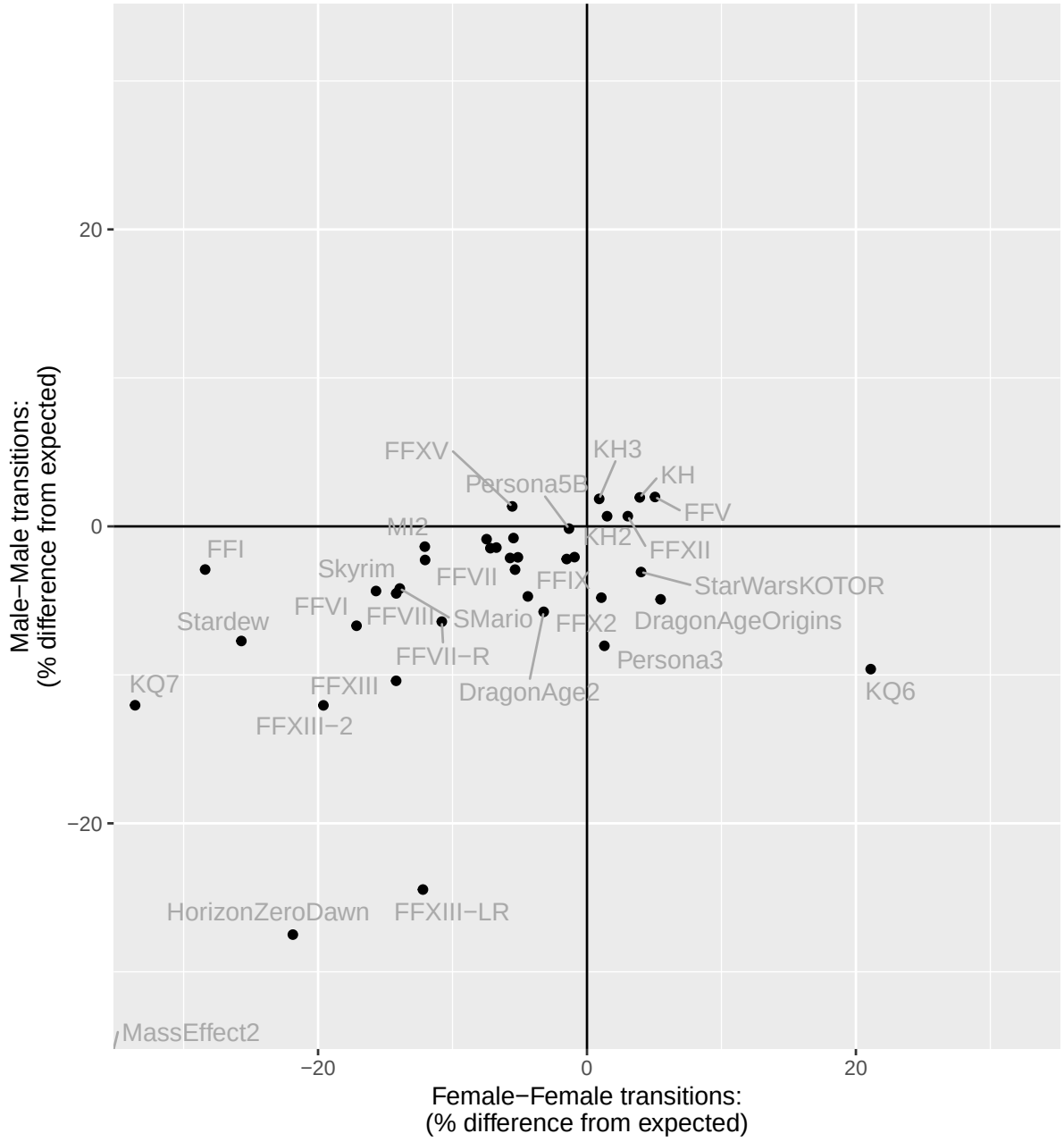
Write the data:

```
write.csv(permutationResults, "../results/transitionsPermutationTest.csv",row.names = F)
```

Plot difference from expected

```
permutationResults$shortName2 = permutationResults$shortName
permutationResults$shortName2 = gsub("KingdomHearts","KH",permutationResults$shortName2)
permutationResults$shortName2 = gsub("KingsQuest","KQ",permutationResults$shortName2)
permutationResults$shortName2 = gsub("_Remake","-R",permutationResults$shortName2)
permutationResults$shortName2 = gsub("_B","",permutationResults$shortName2)
permutationResults$shortName2 = gsub("_DS","",permutationResults$shortName2)
permutationResults$shortName2[permutationResults$shortName2=="TheSecretOfMonkeyIsland"] = "MI1"
permutationResults$shortName2[permutationResults$shortName2=="MonkeyIsland2"] = "MI2"
permutationResults$shortName2[permutationResults$shortName2=="TheCurseOfMonkeyIsland"] = "MI3"
permutationResults$shortName2[permutationResults$shortName2=="SuperMarioRPG"] = "SMario"
permutationResults$shortName2[permutationResults$shortName2=="StardewValley"] = "Stardew"
permutationResults$shortName2[permutationResults$shortName2=="FFX_B"] = "FFX"

permResPlot =   ggplot(permutationResults,
    aes(x=diffExpEmp.f2f*100,y=diffExpEmp.m2m*100)) +
    geom_point() +
    coord_cartesian(ylim=c(-32,32),xlim=c(-32,32))+
    ylab("Male-Male transitions:\n(% difference from expected)") +
    geom_hline(yintercept = 0) + geom_vline(xintercept = 0) +
    xlab("Female-Female transitions:\n(% difference from expected)") +
    geom_text_repel(aes(label=shortName2),color="dark gray",force = 10)
permResPlot
```

```
## Warning: ggrepel: 9 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```




```
pdf('../results/graphs/transitions/Transitions_DiffFromExpected.pdf', width=6,height=6)
permResPlot
```

```
## Warning: ggrepel: 12 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

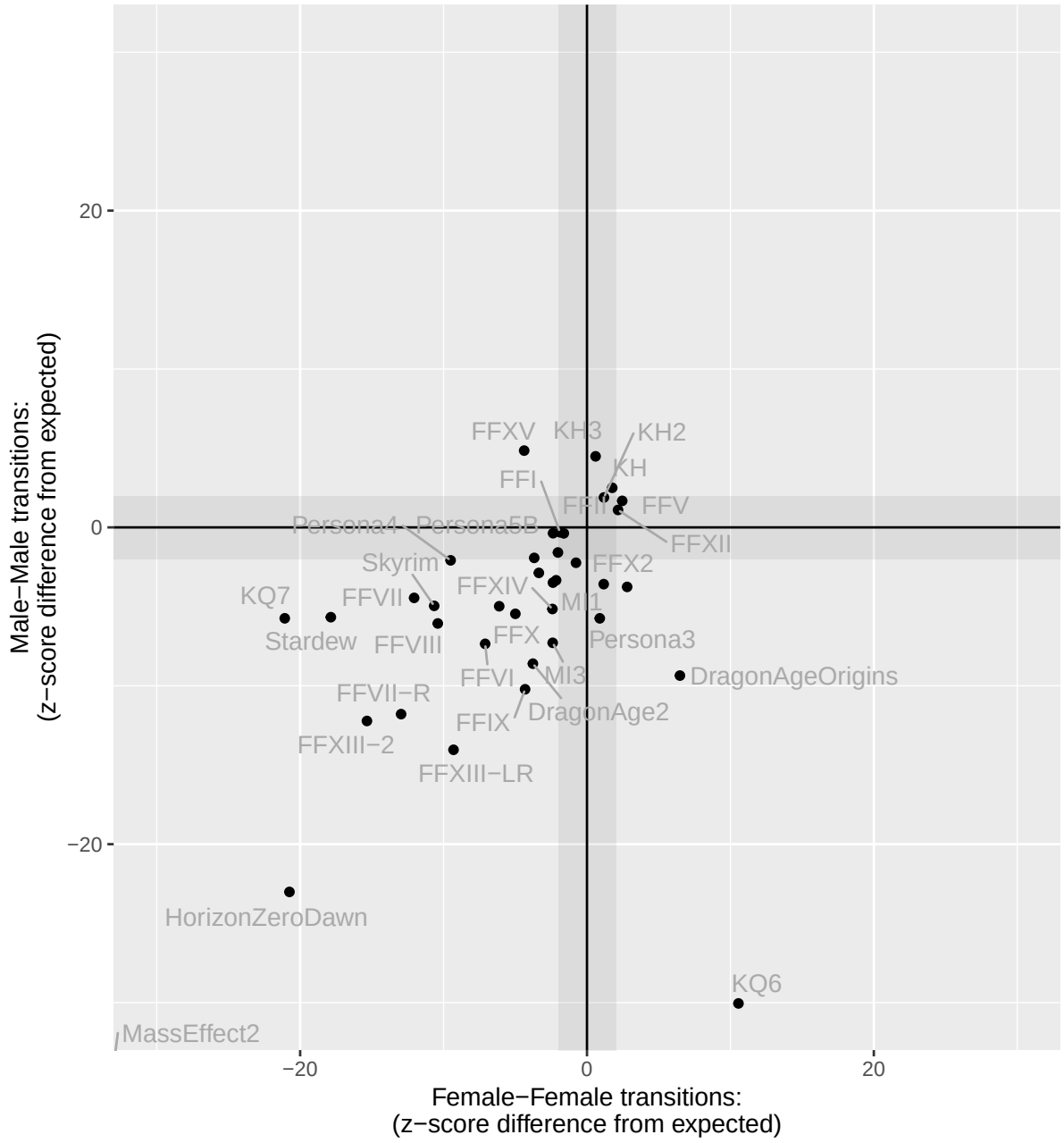
```
dev.off()
```

```
## pdf
##   2
```

```
  ggplot(permutationResults,aes(x=f2f.z,y=m2m.z)) +
    annotate("rect", xmin = -1000, xmax = 1000, ymin = -2, ymax = 2, alpha = .1) +
    annotate("rect", xmin = -2, xmax = 2, ymin = -1000, ymax = 1000, alpha = .1) +
    geom_point() +
    coord_cartesian(ylim=c(-30,30),xlim=c(-30,30))+
    ylab("Male-Male transitions:\n(z-score difference from expected)") +
    geom_hline(yintercept = 0) + geom_vline(xintercept = 0) +
```

```
    xlab("Female-Female transitions:\n(z-score difference from expected)") +
    geom_text_repel(aes(label=shortName2),color="dark gray",force = 10)
```

```
## Warning: ggrepel: 7 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```




```
xs = c(-20,-10,0,10,20)
ls = log(100+xs)
permResultsZScore = ggplot(permutationResults,aes(x=log(100+f2f.z),y=log(100+m2m.z))) +
    annotate("rect", xmin = -1000, xmax = 1000, ymin = log(100-2), ymax = log(102), alpha = .1) +
    annotate("rect", xmin = log(100-2), xmax = log(102), ymin = -1000, ymax = 1000, alpha = .1) +
    geom_point() +
    coord_cartesian(ylim=c(4.25,4.9),xlim=c(4.25,4.9))+
    ylab("Male-Male transitions:\n(z-score difference from expected)") +
 #   geom_hline(yintercept = 0) + geom_vline(xintercept = 0) +
    xlab("Female-Female transitions:\n(z-score difference from expected)") +
    geom_text_repel(aes(label=shortName2),color="dark gray",force = 10) +
```

```
    scale_x_continuous(breaks=ls,labels=xs,
       sec.axis = sec_axis(~.*1,
                           breaks = ls,
                           labels=c("Fewer female-female transitions\nthan expected","",
                                    "As expected","",
                                    "More female-female transitions\nthan expected")))+
  scale_y_continuous(breaks=ls,labels=xs) +
    theme(panel.grid.minor = element_blank())



pdf("../results/graphs/transitions/Transitions_DiffFromExpected_ZScores.pdf",width=6.5,height=6)
permResultsZScore
```

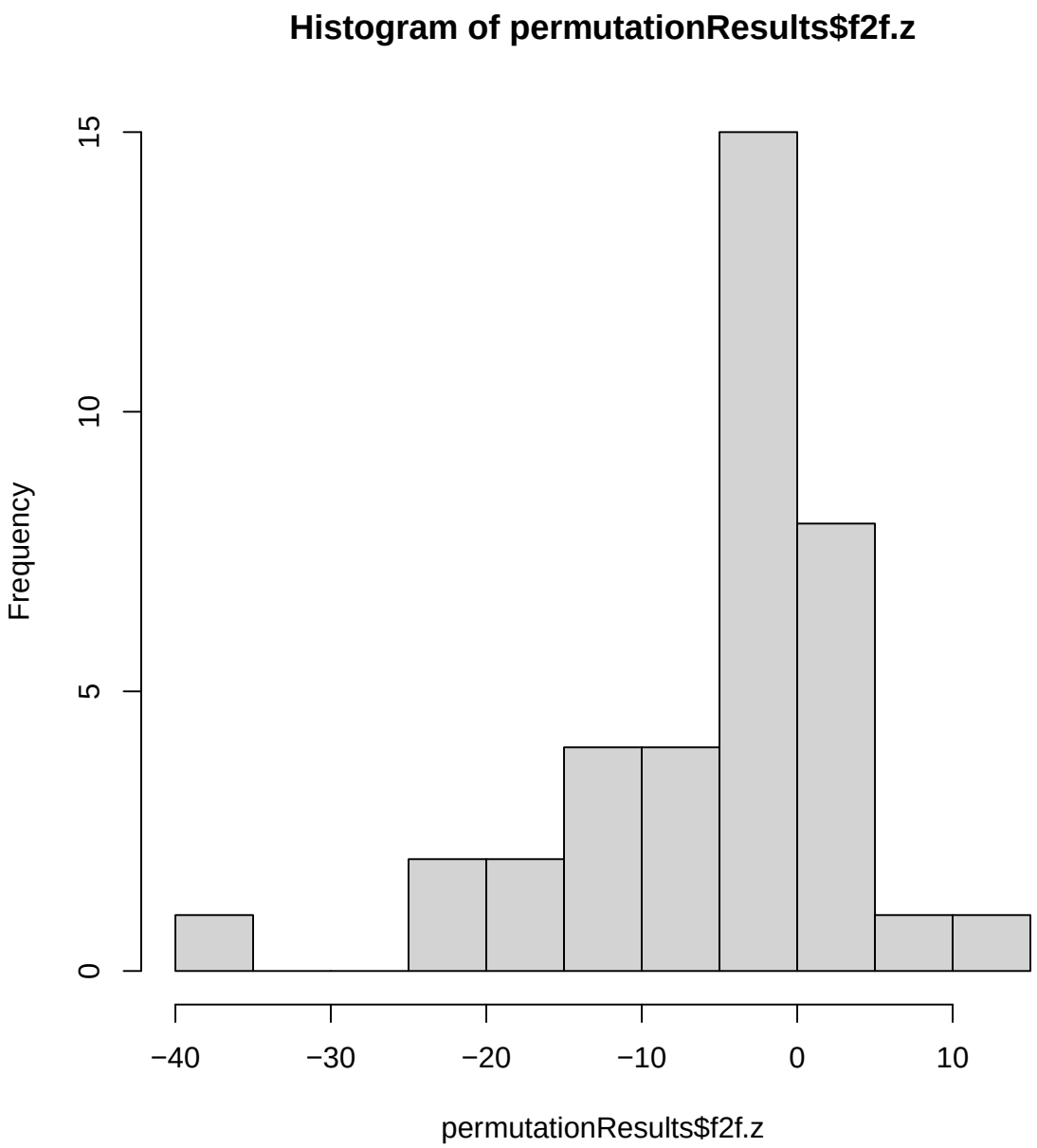
```
## Warning: ggrepel: 17 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

```
dev.off()
```

```
## pdf
##   2
```

In general, transitions within genders is lower than expected, while transitions between genders is higher than expected.

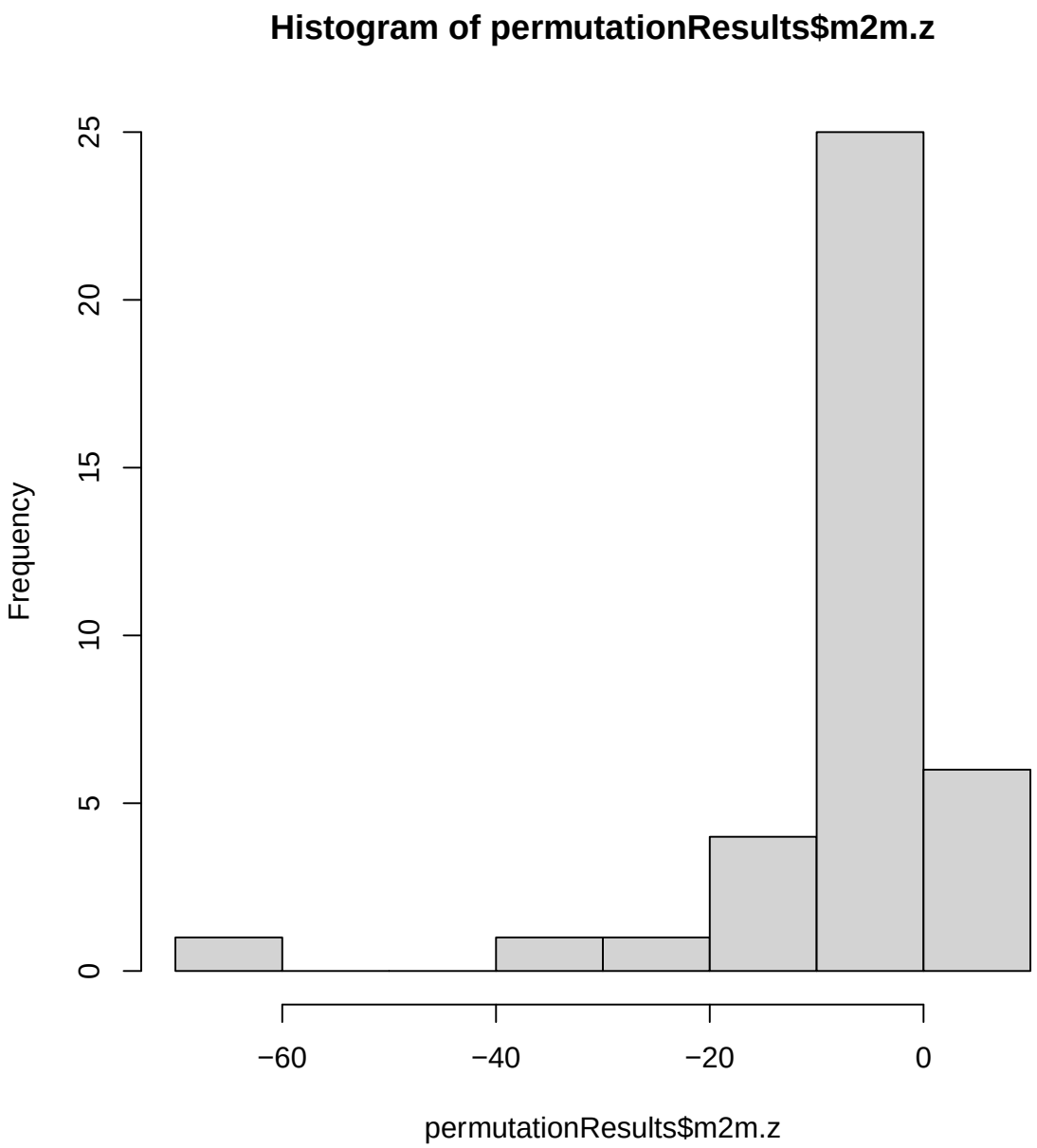
```
hist(permutationResults$f2f.z)
```




```
mean(permutationResults$f2f.z)
```

```
## [1] -5.348596
```

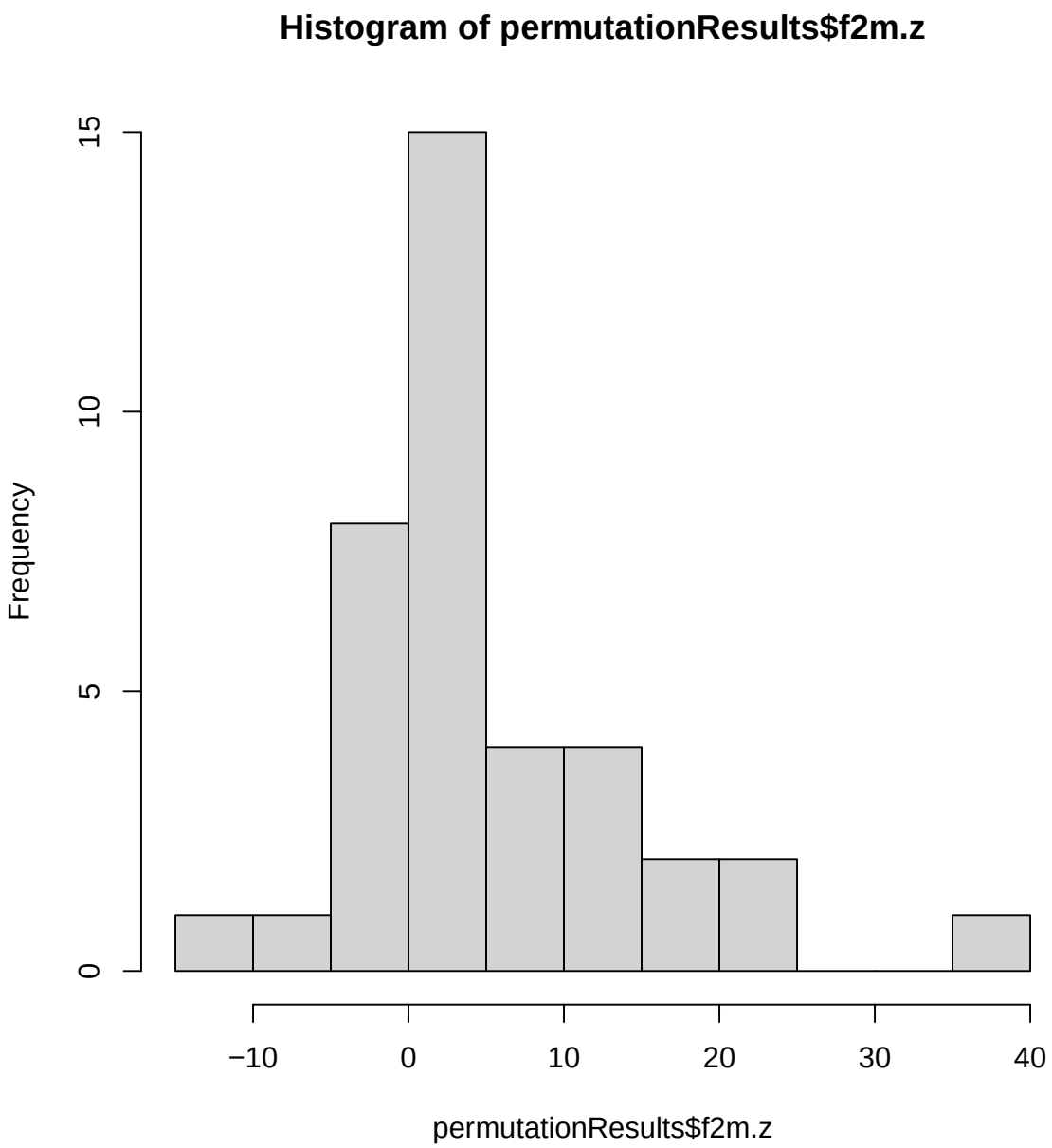
```
hist(permutationResults$m2m.z)
```




```
mean(permutationResults$m2m.z)
```

```
## [1] -6.826839
```

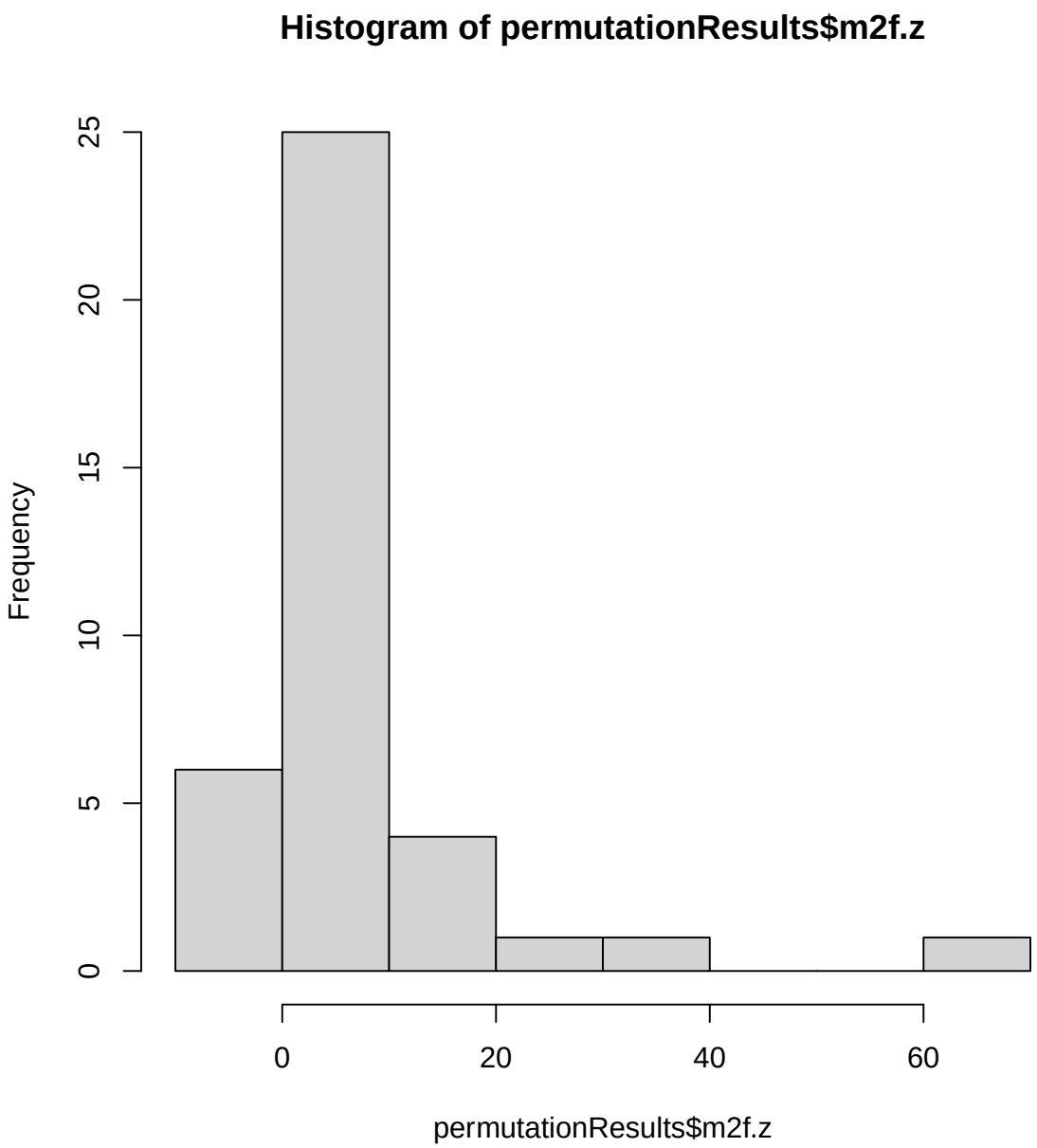
```
hist(permutationResults$f2m.z)
```




```
mean(permutationResults$f2m.z)
```

```
## [1] 5.348596
```

```
hist(permutationResults$m2f.z)
```




```
mean(permutationResults$m2f.z)
```

```
## [1] 6.826839
```

Number of games with within lower than expected female-to-female transitions:

```
lowF2F = table(permutationResults$f2f.z < 0 &
                 permutationResults$f2f.p<0.05)
lowM2M = table(permutationResults$m2m.z < 0 &
                 permutationResults$m2m.p<0.05)
lowF2F
```

```
##
## FALSE  TRUE
##    13    25
```

```
lowM2M
```

```
##
```

```
## FALSE  TRUE
##    10    28
```

25 games had significantly lower female-to-female transitions than expected by chance, and 25 games had significantly lower male-to-male transitions than expected by chance.

```
binom.test(table(permutationResults$f2f.z < permutationResults$m2m.z))
```

```
##
##  Exact binomial test
##
## data:  table(permutationResults$f2f.z < permutationResults$m2m.z)
## number of successes = 19, number of trials = 38, p-value = 1
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.333789 0.666211
## sample estimates:
## probability of success
##                    0.5
```

```
# Some imbalances:
table(permutationResults$m2m.z < 0 & permutationResults$m2m.p<0.05,
      permutationResults$f2f.z < 0 & permutationResults$f2f.p<0.05)
```

```
##
##         FALSE TRUE
##   FALSE     7    3
##   TRUE      6   22
```

```
# Game with less m-m and more f-f than expected:
permutationResults[(permutationResults$m2m.z <0 &
                      permutationResults$m2m.p<0.05) &
                     (permutationResults$f2f.z >0 &
                        permutationResults$f2f.p<0.05),]
```

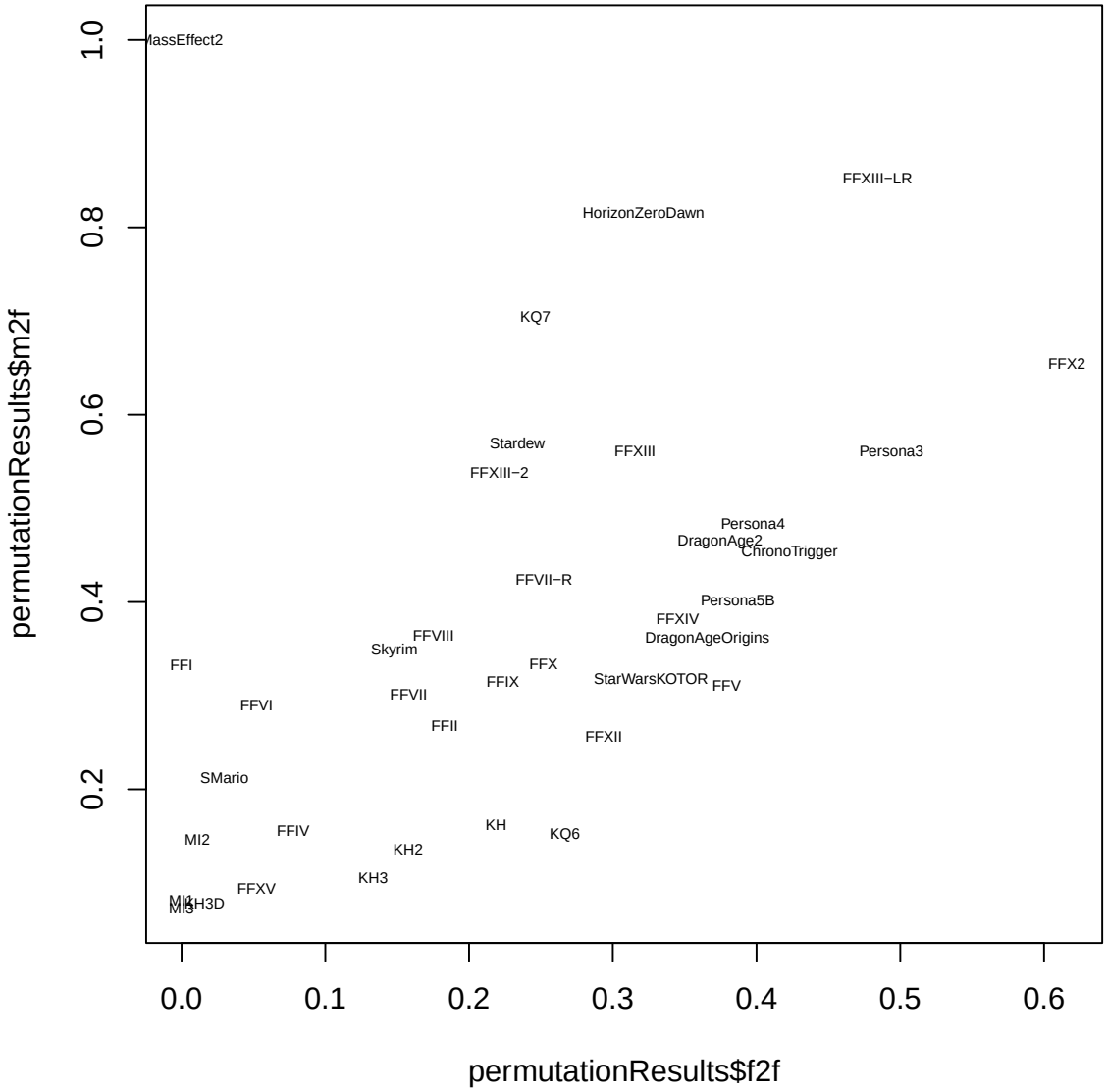
```
##                                  folder                                    series
## 3  ../data/DragonAge/DragonAgeOrigins_B                                Dragon Age
## 27       ../data/KingsQuest/KingsQuest6                               King's Quest
## 37  ../data/StarWarsKOTOR/StarWarsKOTOR Star Wars: Knights of the Old Republic
##                                     game          shortName       m2m
## 3                    Dragon Age: Origins DragonAgeOrigins_B 0.6392202
## 27                       King's Quest VI        KingsQuest6 0.8475798
## 37 Star Wars: Knights of the Old Republic      StarWarsKOTOR 0.6820782
##          f2m       m2f       f2f m2m.raw f2m.raw m2f.raw f2f.raw  m2m.mean
## 3  0.6341757 0.3607798 0.3658243    2787    1581    1573     912 0.6883674
## 27 0.7333333 0.1524202 0.2666667     823     110     148      40 0.9437190
## 37 0.6734463 0.3179218 0.3265537    1431     596     667     289 0.7128301
##         m2m.z m2m.p  f2m.mean      f2m.z f2m.p   m2f.mean     m2f.z m2f.p
## 3   -9.356370 0.001 0.6889318  -6.481471 0.001 0.31163261  9.356370 0.001
## 27 -30.054323 0.001 0.9443704 -10.561529 0.001 0.05628103 30.054323 0.001
## 37  -3.757067 0.001 0.7136756  -2.800403 0.004 0.28716987  3.757067 0.001
##      f2f.mean     f2f.z f2f.p diffExpEmp.m2m diffExpEmp.f2f       shortName2
## 3  0.31106823  6.481471 0.001    -0.04914720     0.05475608 DragonAgeOrigins
## 27 0.05562956 10.561529 0.001    -0.09613916     0.21103710              KQ6
## 37 0.28632442  2.800403 0.004    -0.03075196     0.04022926    StarWarsKOTOR
```

```
# Game with less f-f and more m-m than expected:
permutationResults[(permutationResults$m2m.z >0 &
                      permutationResults$m2m.p<0.05) &
                     (permutationResults$f2f.z <0 &
                        permutationResults$f2f.p<0.05),]
```

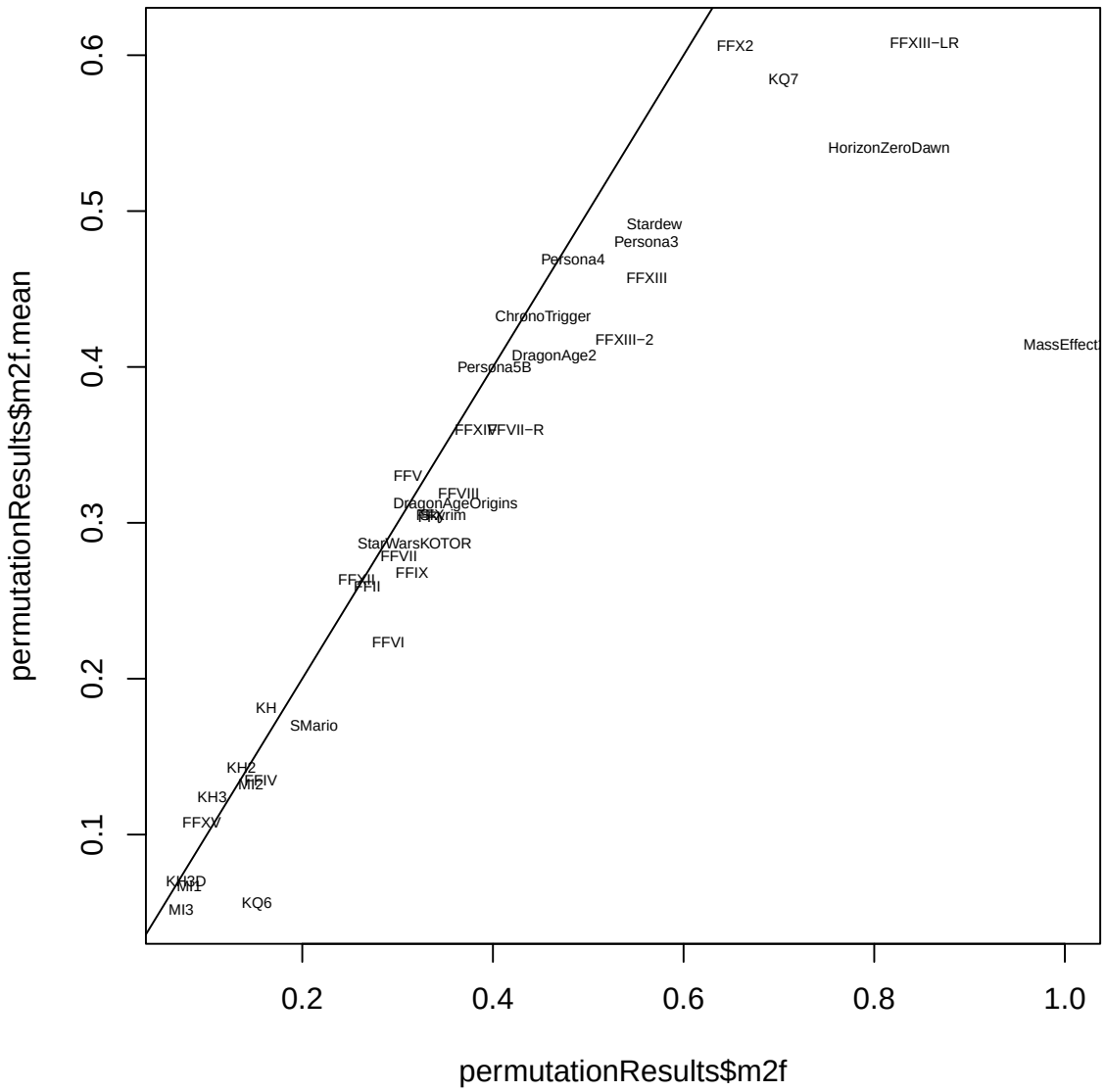
```
##                       folder        series             game shortName      m2m
## 21 ../data/FinalFantasy/FFXV Final Fantasy Final Fantasy XV      FFXV 0.905426
##          f2m        m2f        f2f m2m.raw f2m.raw m2f.raw f2f.raw  m2m.mean
## 21 0.9477612 0.09457398 0.05223881    4155     381     434      21 0.8919802
##       m2m.z m2m.p  f2m.mean   f2m.z f2m.p  m2f.mean     m2f.z m2f.p  f2f.mean
## 21 4.849541 0.001 0.8921863 4.37991 0.001 0.1080198 -4.849541 0.001 0.1078137
##       f2f.z f2f.p diffExpEmp.m2m diffExpEmp.f2f shortName2
## 21 -4.37991 0.001     0.01344582    -0.05557492       FFXV
```

The relationship between the different transition types is correlated, but there is some variation:

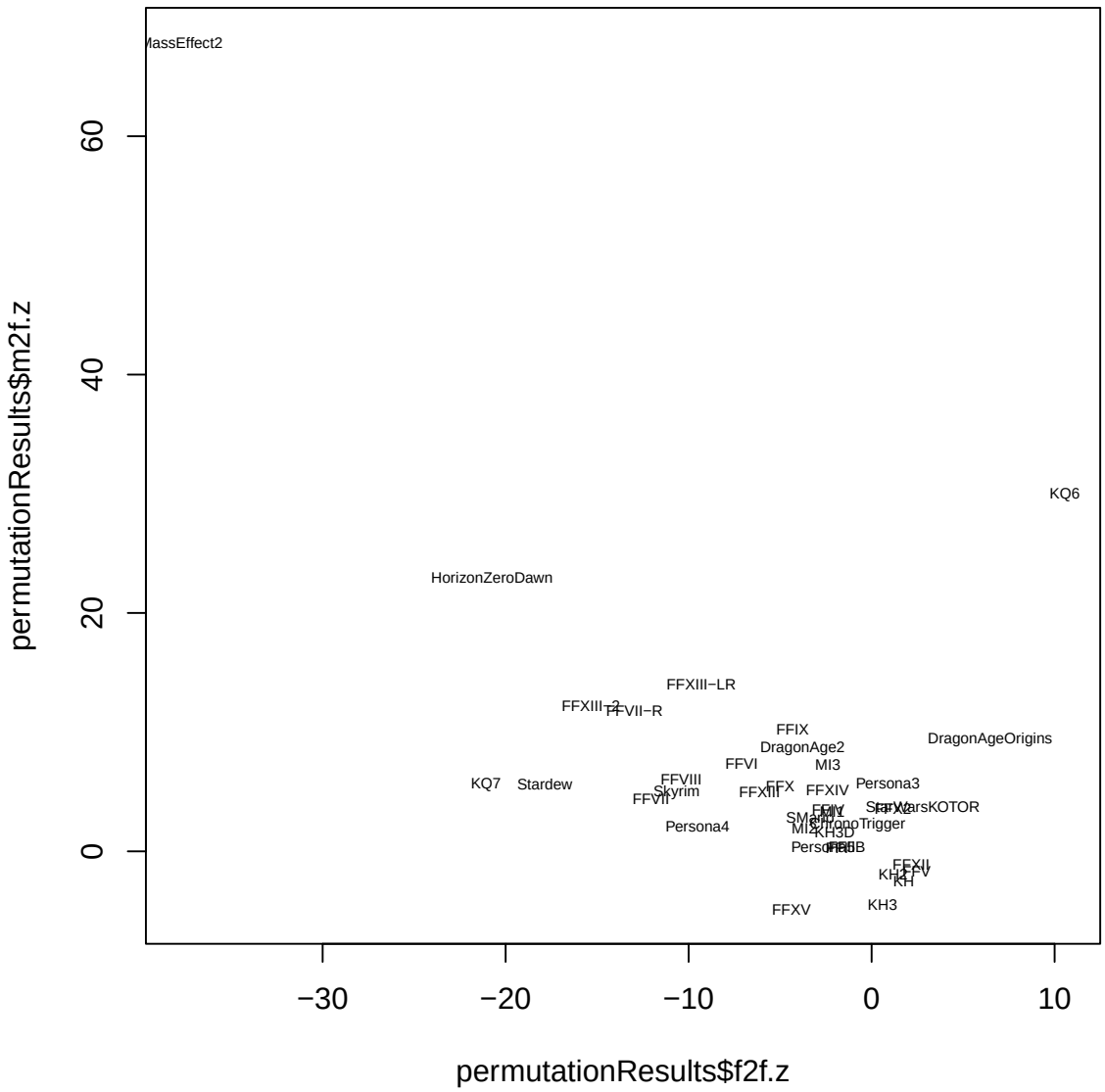
```
plot(permutationResults$f2f,permutationResults$m2f,col=NA)
text(permutationResults$f2f,permutationResults$m2f,permutationResults$shortName2,cex=0.5)
```



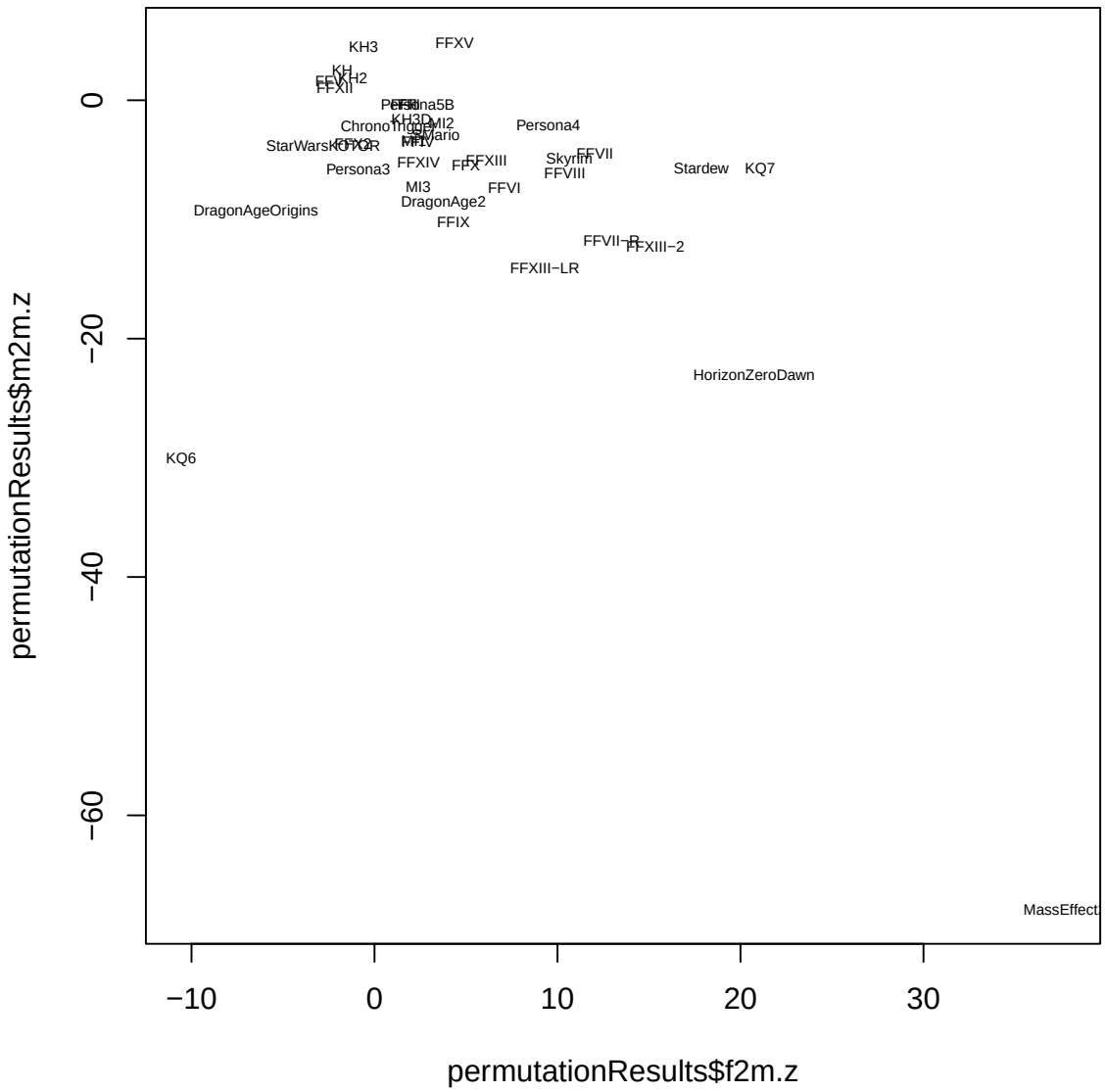

```
plot(permutationResults$m2f,permutationResults$m2f.mean,col=NA)
text(permutationResults$m2f,permutationResults$m2f.mean,permutationResults$shortName2,cex=0.5)
abline(0,1)
```



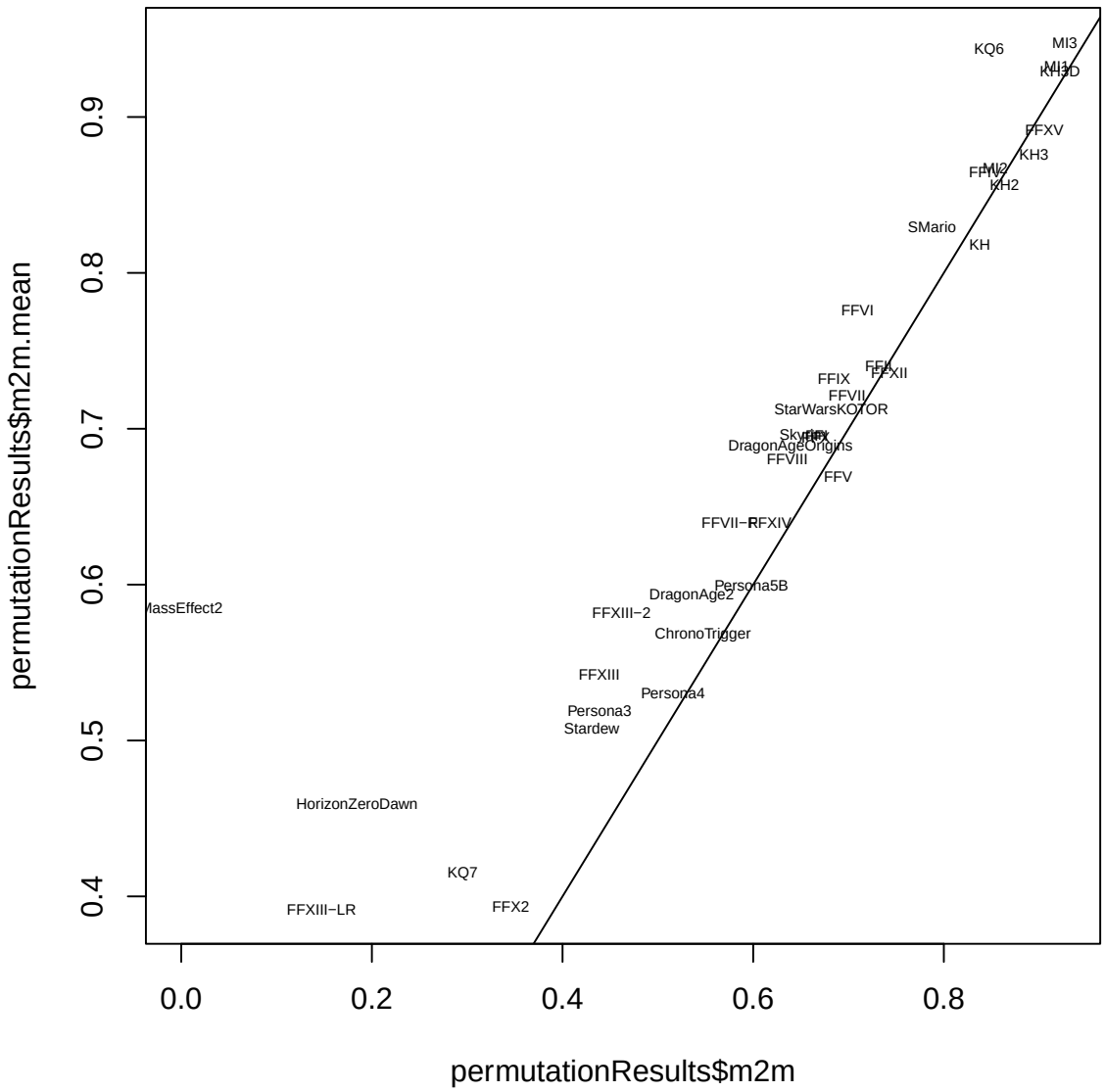

```
plot(permutationResults$f2f.z,permutationResults$m2f.z,col=NA)
text(permutationResults$f2f.z,permutationResults$m2f.z,permutationResults$shortName2,cex=0.5)
```




```
plot(permutationResults$f2m.z,permutationResults$m2m.z,col=NA)
text(permutationResults$f2m.z,permutationResults$m2m.z,permutationResults$shortName2,cex=0.5)
```




```
plot(permutationResults$m2m,permutationResults$m2m.mean,col=NA)
text(permutationResults$m2m,permutationResults$m2m.mean,permutationResults$shortName2,cex=0.5)
abline(0,1)
```



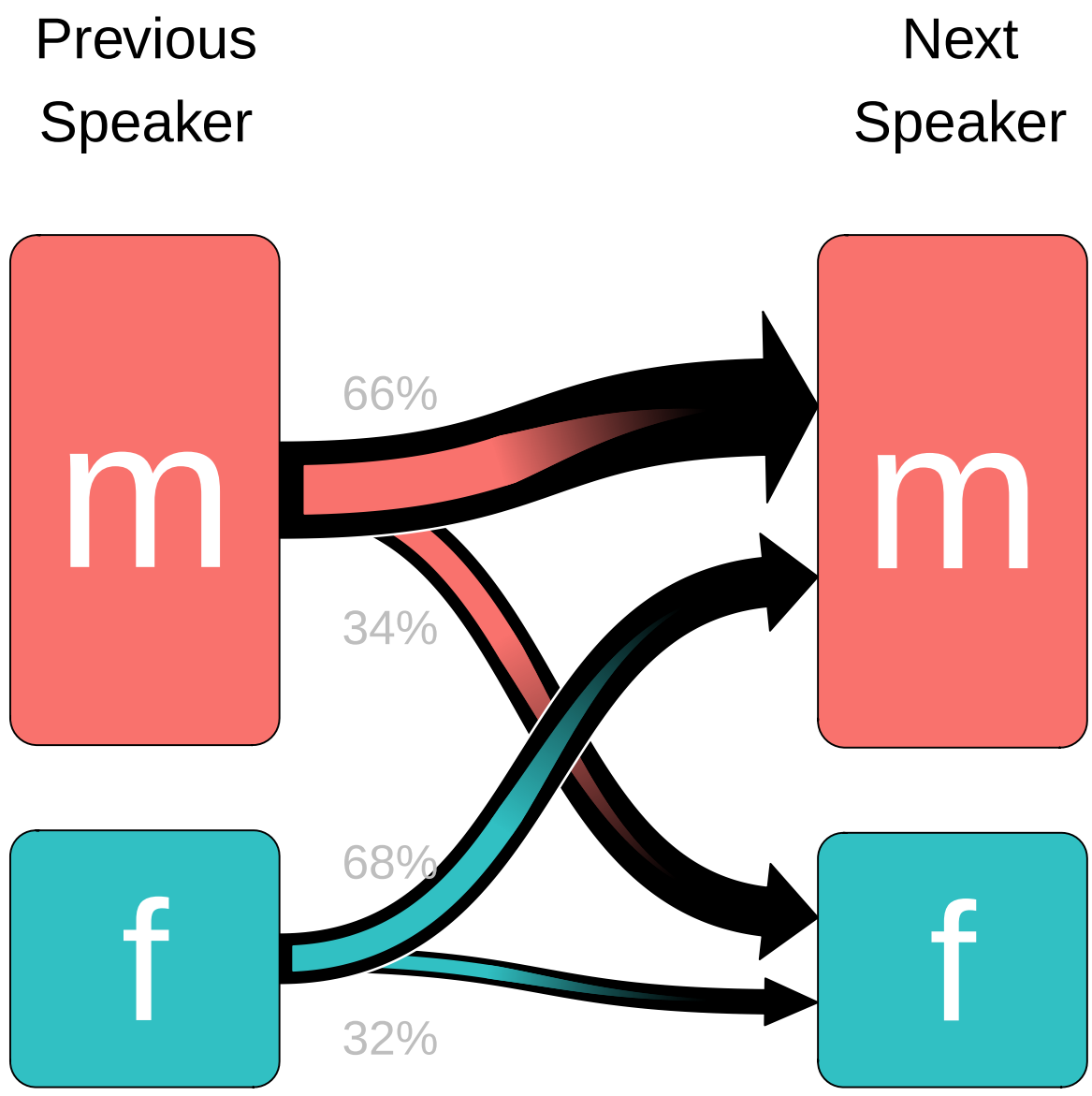
Fancy class for plotting transitions:

```
transitions <- trueTransitions.AllGames.Raw %>%
  getRefClass("Transition")$new(
      label=c("Previous\nSpeaker", "Next\nSpeaker"),
      skip_shadows = TRUE,
      min_lwd = unit(0.1, "mm"), max_lwd = unit(14, "mm"),
      box_label_cex = 2,
      fill_clr = list(c("#f9726d","#31c0c3"),c("#f9726d","#31c0c3")))
```

Plots for all games

Visualisation of overall transitions for all games:

```
transitions$render()
grid::grid.text(label=paste0(round(100*trueTransitions.AllGames),"%"),
                x=c(0.35,0.35,0.35,0.35),
                y=c(0.6,0.2, 0.4,0.05),
                gp=gpar(fontsize=20, col="grey"))
```

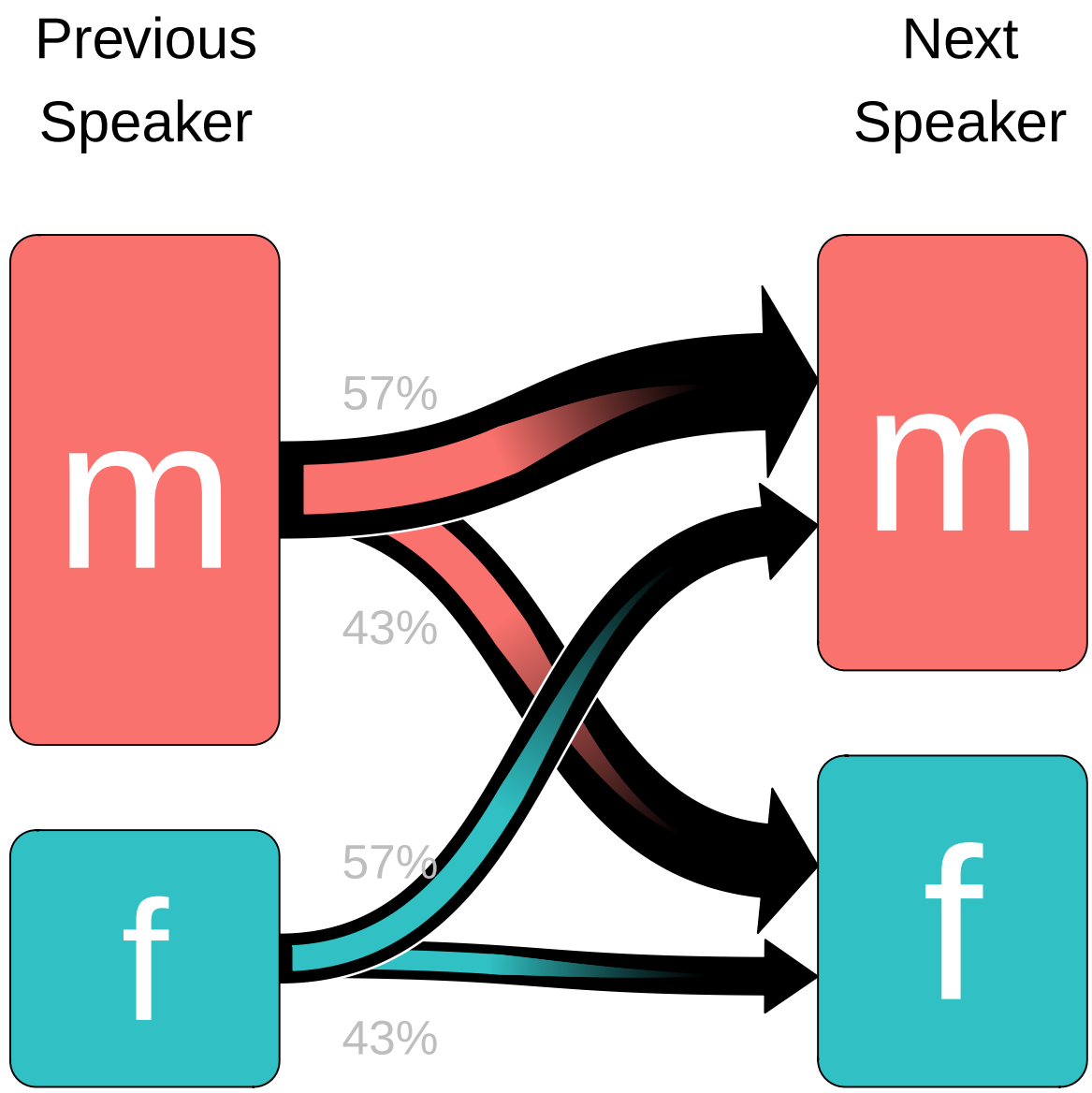



```
pdf("../results/graphs/transitions/Transitions.pdf")
transitions$render()
grid::grid.text(label=paste0(round(100*trueTransitions.AllGames),"%"),
                x=c(0.35,0.35,0.35,0.35),
                y=c(0.6,0.2, 0.4,0.05),
                gp=gpar(fontsize=20, col="grey"))
dev.off()
```

```
## pdf
##   2
```

Expected proportions under permutation for all games:

```
expectedProp = matrix(c(permutedTransitionStats.AllGames['m2m.mean'], permutedTransitionStats.AllGam
permutedTransitionStats.AllGames['f2m.mean'], permutedTransitionStats.AllGames['f2f.mean']),nrow=2,b
rownames(expectedProp) = c("m","f")
colnames(expectedProp) = c("m","f")

totalLinesPerGender = rowSums(trueTransitions.AllGames.Raw)
expectedRaw = round(expectedProp * totalLinesPerGender)

expectedTrans <- expectedRaw %>%
  getRefClass("Transition")$new(
    label=c("Previous\nSpeaker", "Next\nSpeaker"),
    skip_shadows = TRUE,
    min_lwd = unit(0.01, "mm"), max_lwd = unit(14, "mm"),
    box_label_cex = 2,
    fill_clr = list(c("#f9726d","#31c0c3"),c("#f9726d","#31c0c3")))

expectedTrans$render()
grid::grid.text(label=paste0(round(100*expectedProp),"%"),
                x=c(0.35,0.35,0.35,0.35),
                y=c(0.6,0.2, 0.4,0.05),
                gp=gpar(fontsize=20, col="grey"))
```




```
pdf("../results/graphs/transitions/Transitions_ExpectedAllGames.pdf")
expectedTrans$render()
grid::grid.text(label=paste0(round(100*expectedProp),"%"),
                x=c(0.35,0.35,0.35,0.35),
                y=c(0.6,0.2, 0.4,0.05),
                gp=gpar(fontsize=20, col="grey"))
dev.off()
```

```
## pdf
##   2
```

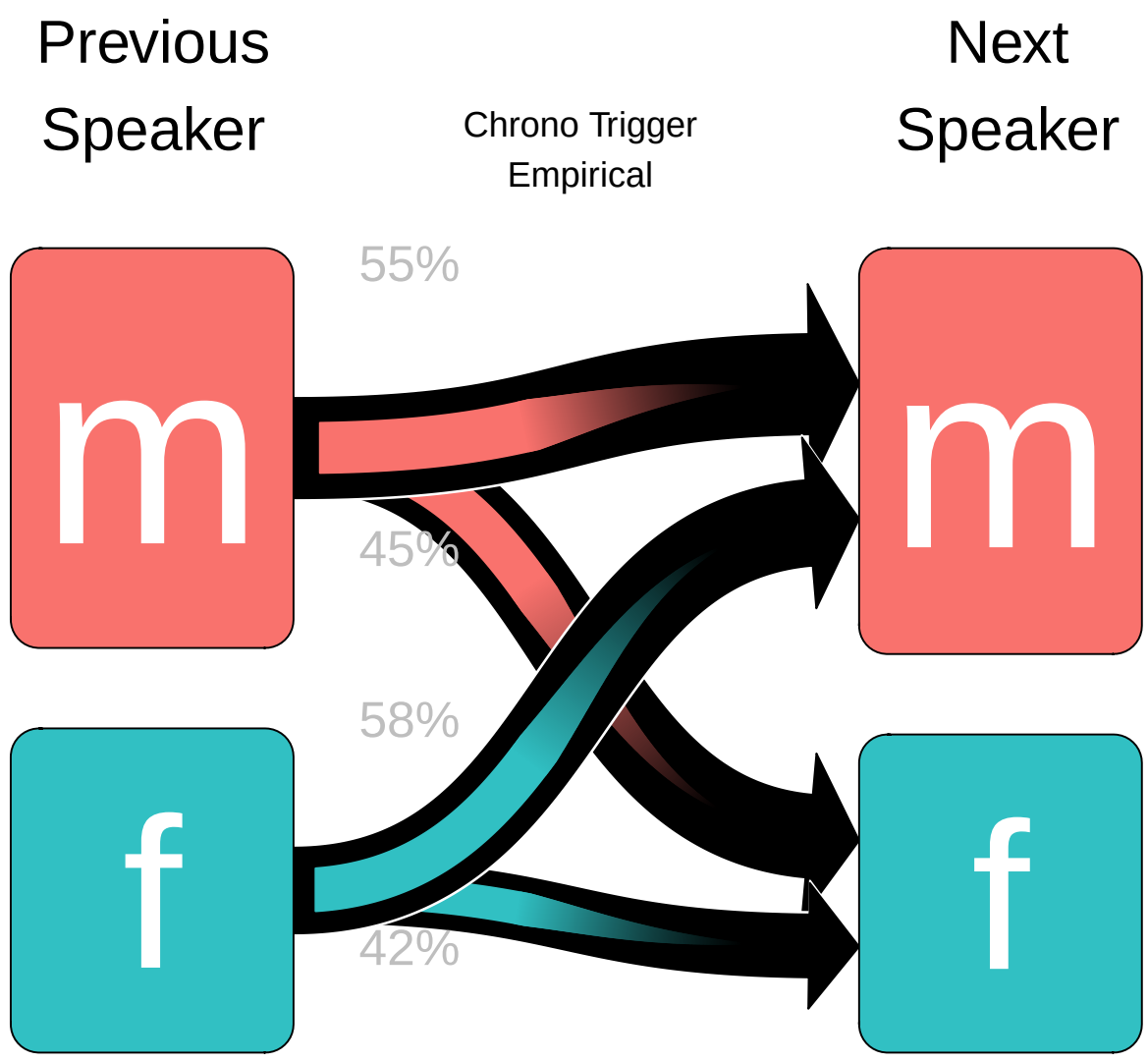
Plots for individual games

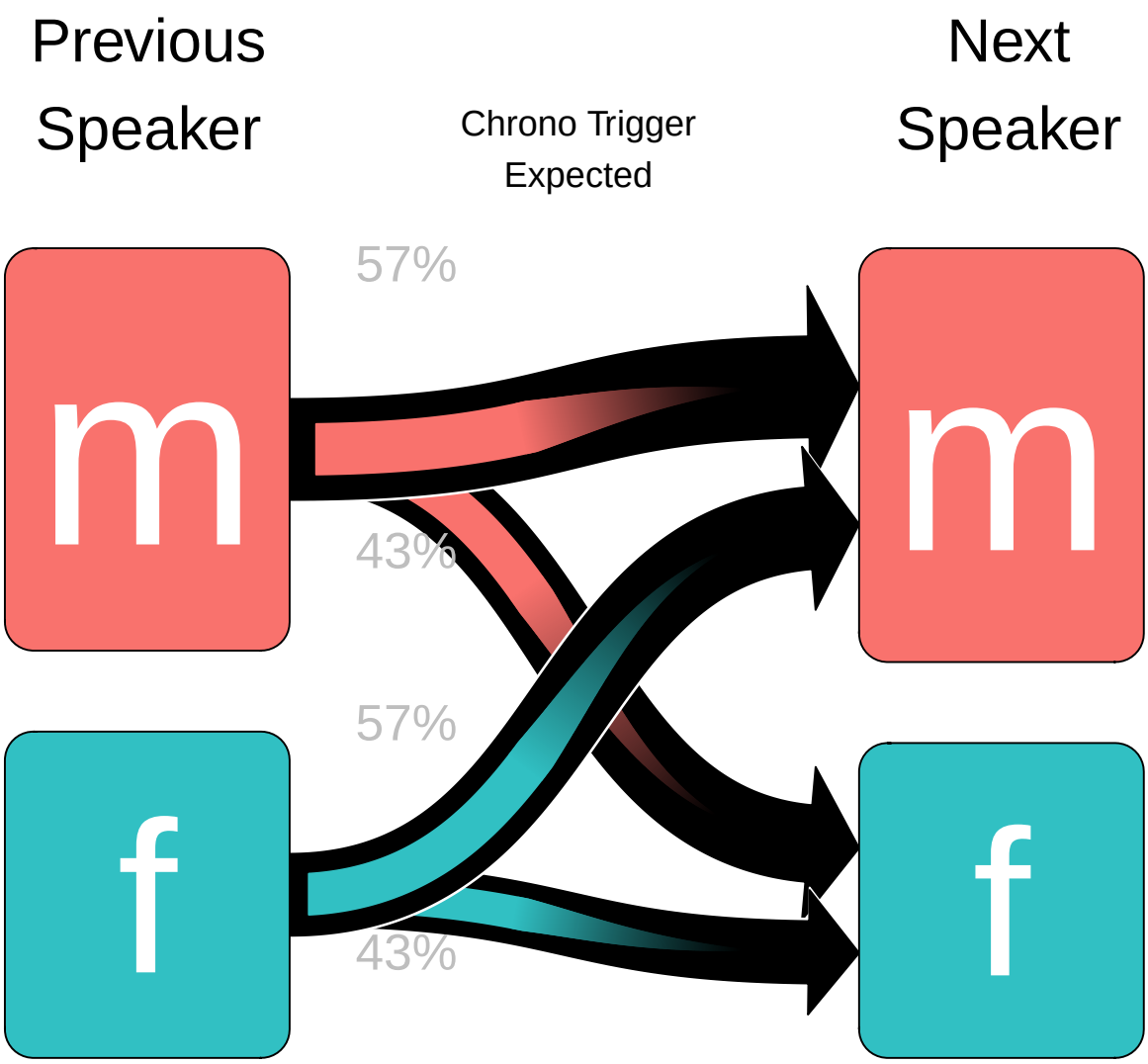
Functions for creating plots from a single game:

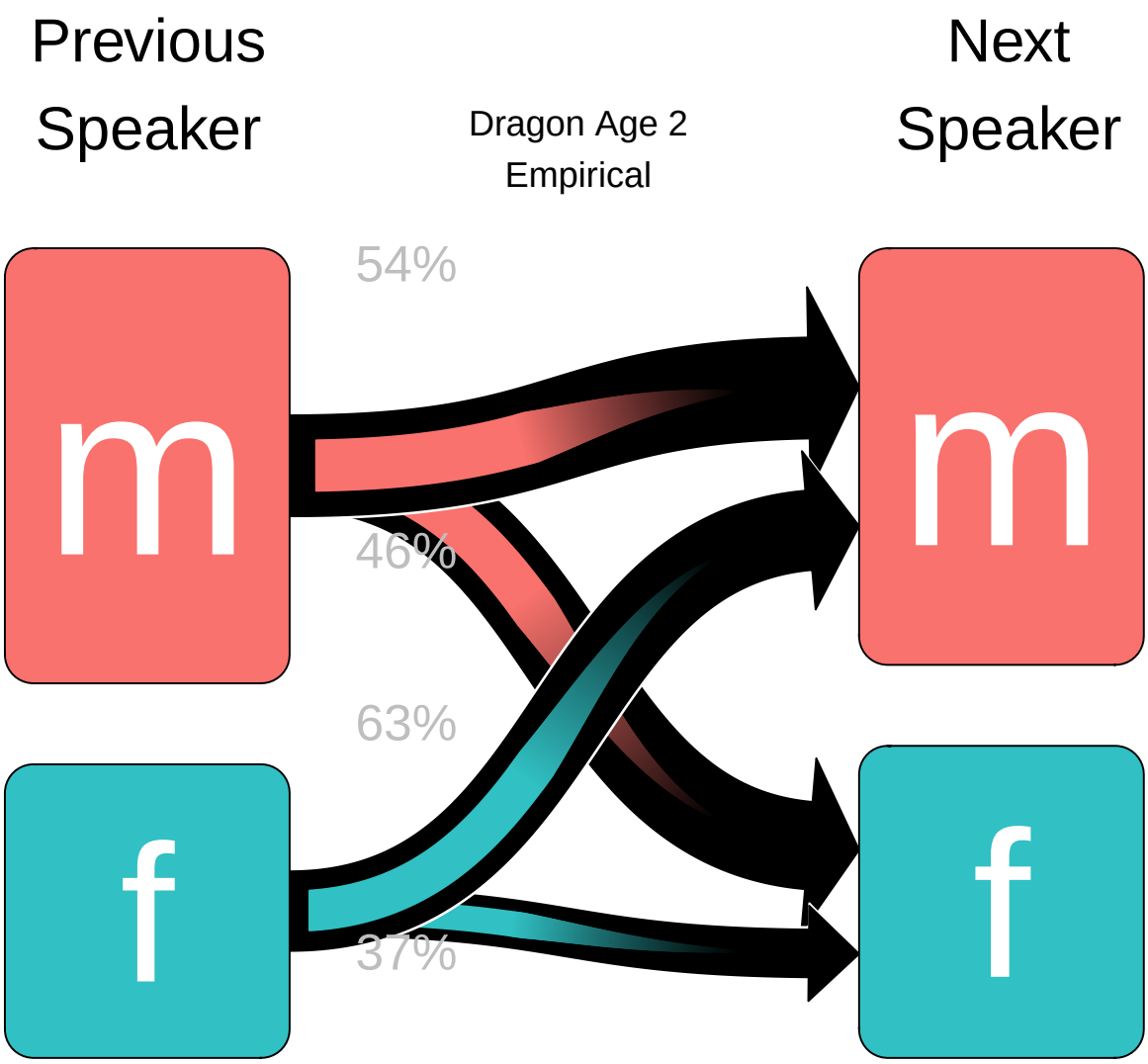
```
plotTransition = function(transProp, transRaw, outFileName,ys,title){
  transitions.ff10 <- transRaw %>%
    getRefClass("Transition")$new(
      label=c("Previous\nSpeaker", "Next\nSpeaker"),
      skip_shadows = TRUE,
      min_lwd = unit(0.01, "mm"), max_lwd = unit(14, "mm"),
      box_label_cex = 2,
      fill_clr = list(c("#f9726d","#31c0c3"),c("#f9726d","#31c0c3")))

  # Plot to console
  transitions.ff10$render()
  grid::grid.text(label=paste0(round(100*transProp),"%"),
                  x=c(0.35,0.35,0.35,0.35),
                  y=ys,
                  gp=gpar(fontsize=20, col="grey"))
  grid::grid.text(label=title,x=0.5,y=0.8, gp=gpar(fontsize=14))
  # Plot to pdf
  pdf(outFileName)
  transitions.ff10$render()
  grid::grid.text(label=paste0(round(100*transProp),"%"),
                  x=c(0.35,0.35,0.35,0.35),
                  y=ys,
                  gp=gpar(fontsize=20, col="grey"))
  grid::grid.text(label=title,x=0.5,y=0.8, gp=gpar(fontsize=14))
  dev.off()

}

makeTransitionGraphForOneGame = function(game,outFileName,ys=c(0.7,0.3, 0.45,0.1),title=""){
  # Empirical plot
  dx = permutationResults[permutationResults$game==game,]
  ff10t = matrix(c(dx$m2m,dx$m2f,dx$f2m,dx$f2f),ncol=2,byrow = T)
  rownames(ff10t) = c("m","f")
  colnames(ff10t) = c("m","f")
  ff10t.raw = matrix(c(dx$m2m.raw,dx$m2f.raw,dx$f2m.raw,dx$f2f.raw),ncol=2,byrow = T)
  rownames(ff10t.raw) = c("m","f")
  colnames(ff10t.raw) = c("m","f")


  plotTransition(ff10t,ff10t.raw,outFileName,ys,
                 title = paste(title,"Empirical",sep="\n"))

  # Expected plot
  expectedProp = matrix(c(dx[['m2m.mean']], dx[['m2f.mean']],
                          dx[['f2m.mean']], dx[['f2f.mean']]),nrow=2,byrow = T)
  rownames(expectedProp) = c("m","f")
  colnames(expectedProp) = c("m","f")

  totalLinesPerGender = rowSums(ff10t.raw)
  expectedRaw = round(expectedProp * totalLinesPerGender)
  outFileName2 = gsub("\\.pdf","_Expected.pdf",outFileName)
  plotTransition(expectedProp,expectedRaw,outFileName2,ys,
                 title = paste(title,"Expected",sep="\n"))

}
```

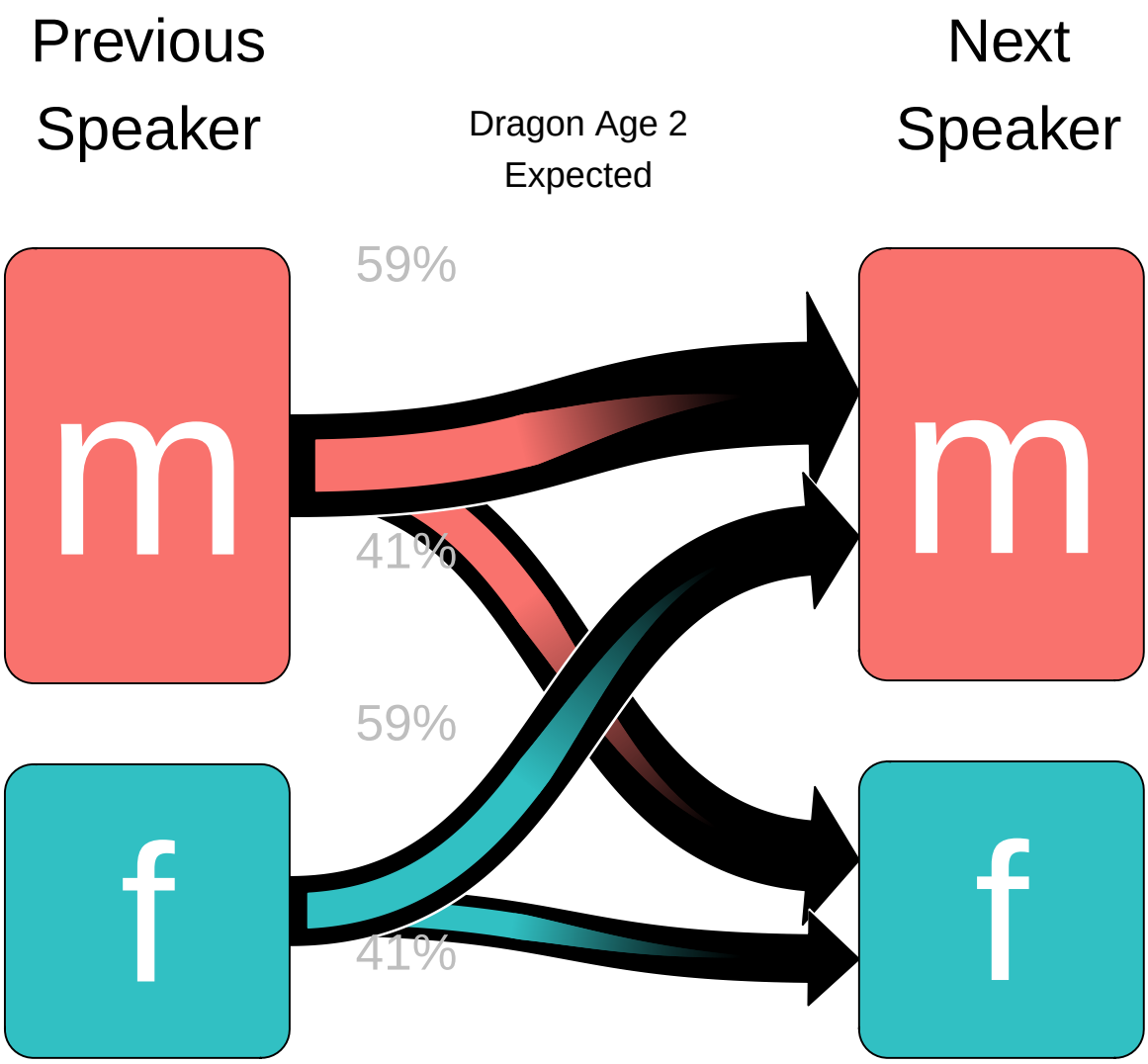
Apply to each game separately:

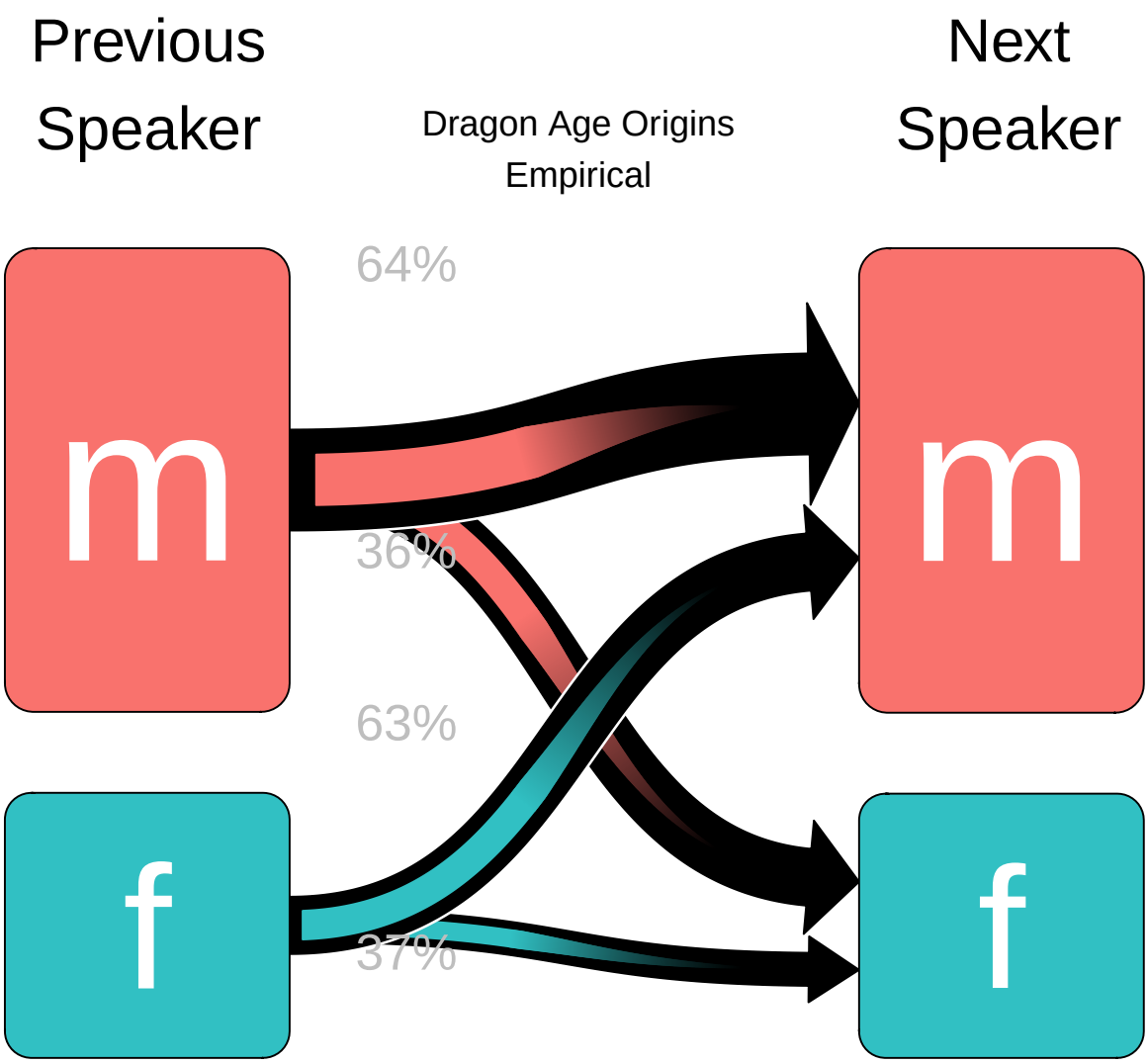
```
for(folder in unique(permutationResults$folder)){
  gameName = permutationResults[permutationResults$folder==folder,]$game[1]
  #print(gameName)
  gameName1 = gsub("[:']","",gameName)
  gameName2 = gsub(" +","_",gameName1)
  pdfFile = paste0("../results/graphs/transitions/Transitions_",gameName2,".pdf")
  if(gameName == "Final Fantasy XV"){
   makeTransitionGraphForOneGame(gameName, pdfFile,ys=c(0.55,0.12, 0.35,0.02),"FF XV")
  } else{
    makeTransitionGraphForOneGame(gameName, pdfFile,title=gameName1)
  }
}
```

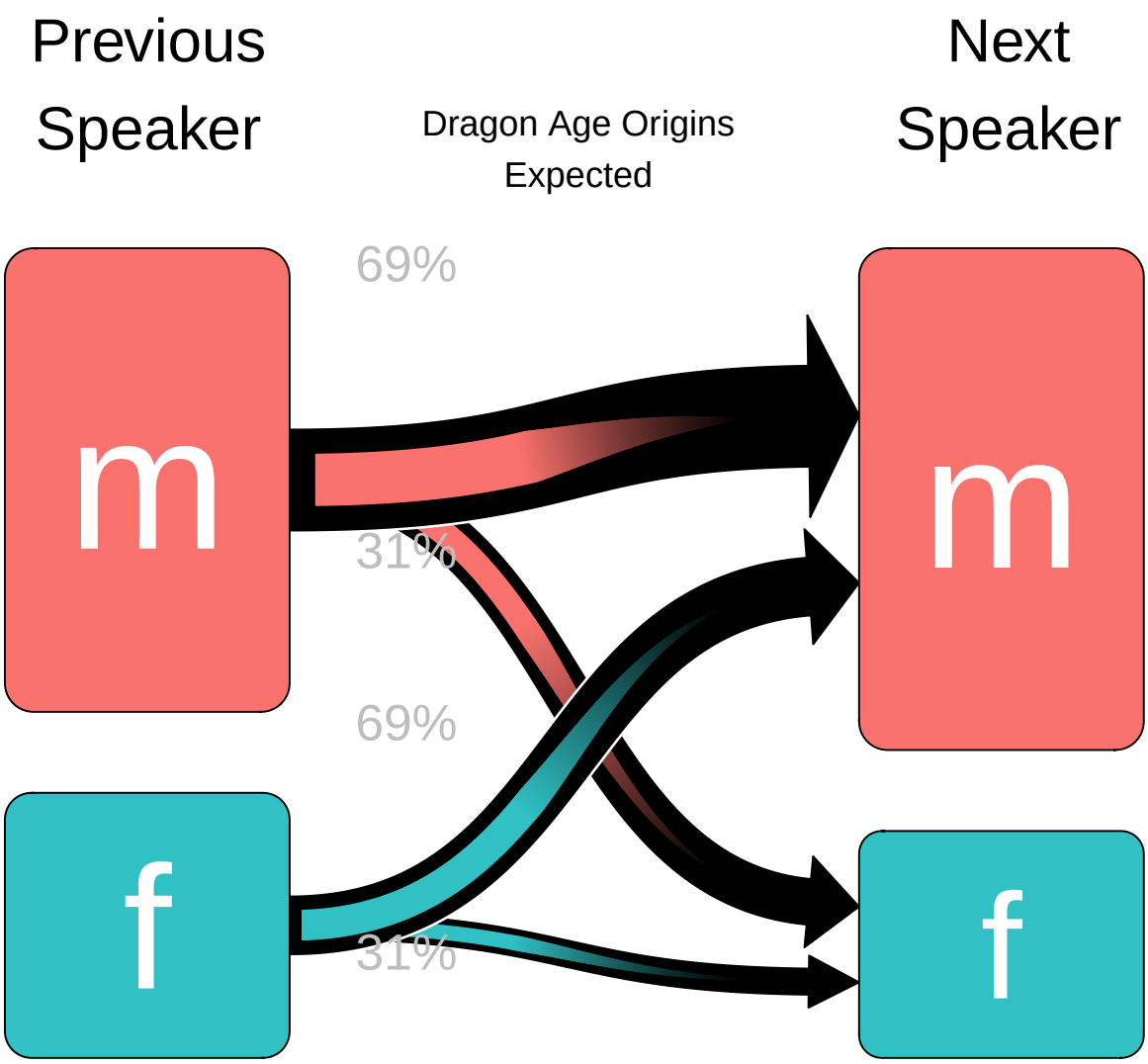


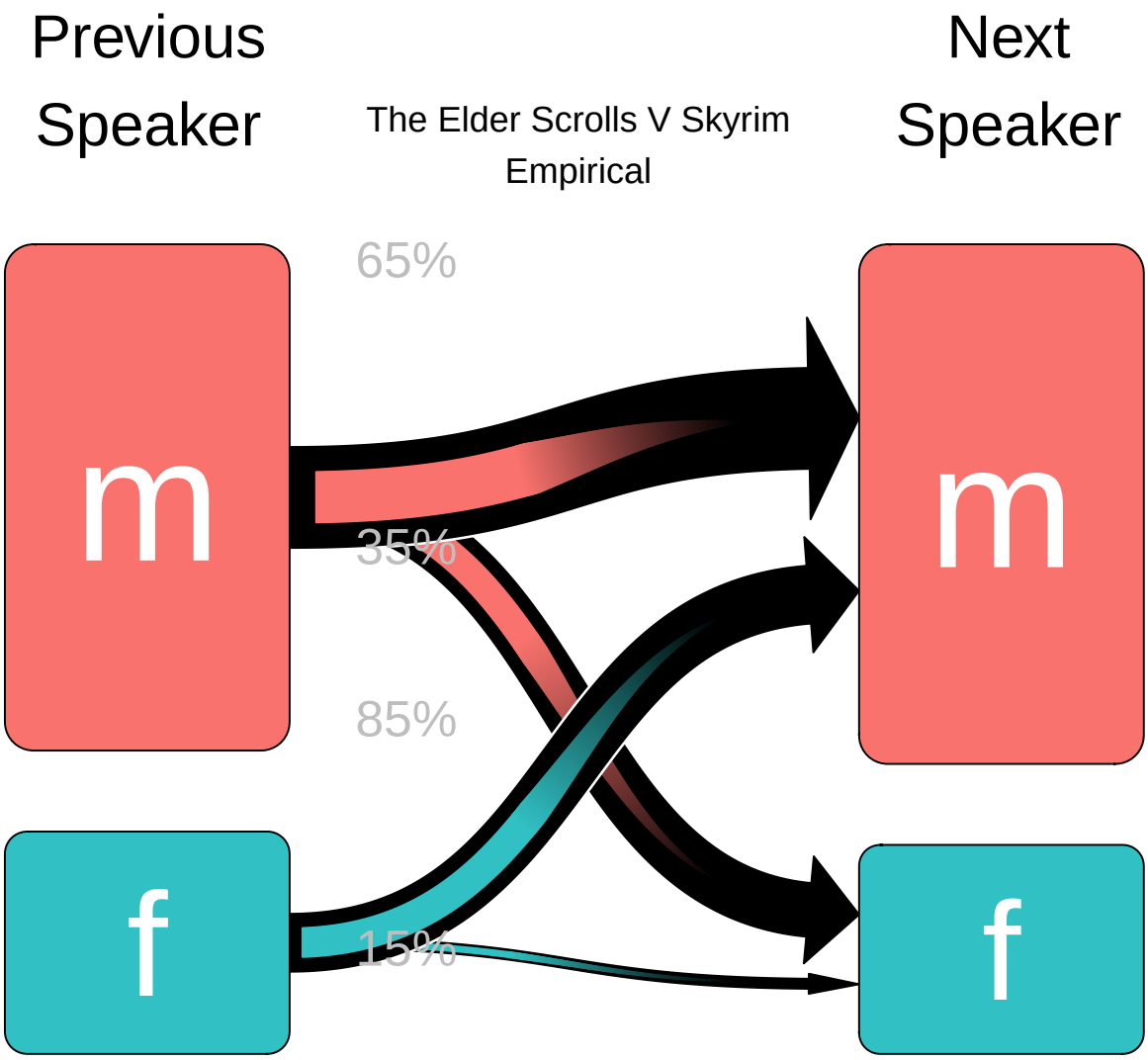

Previous Speaker
Chrono Trigger Expected
Next Speaker
m
f
57%
43%
57%
43%
m
f

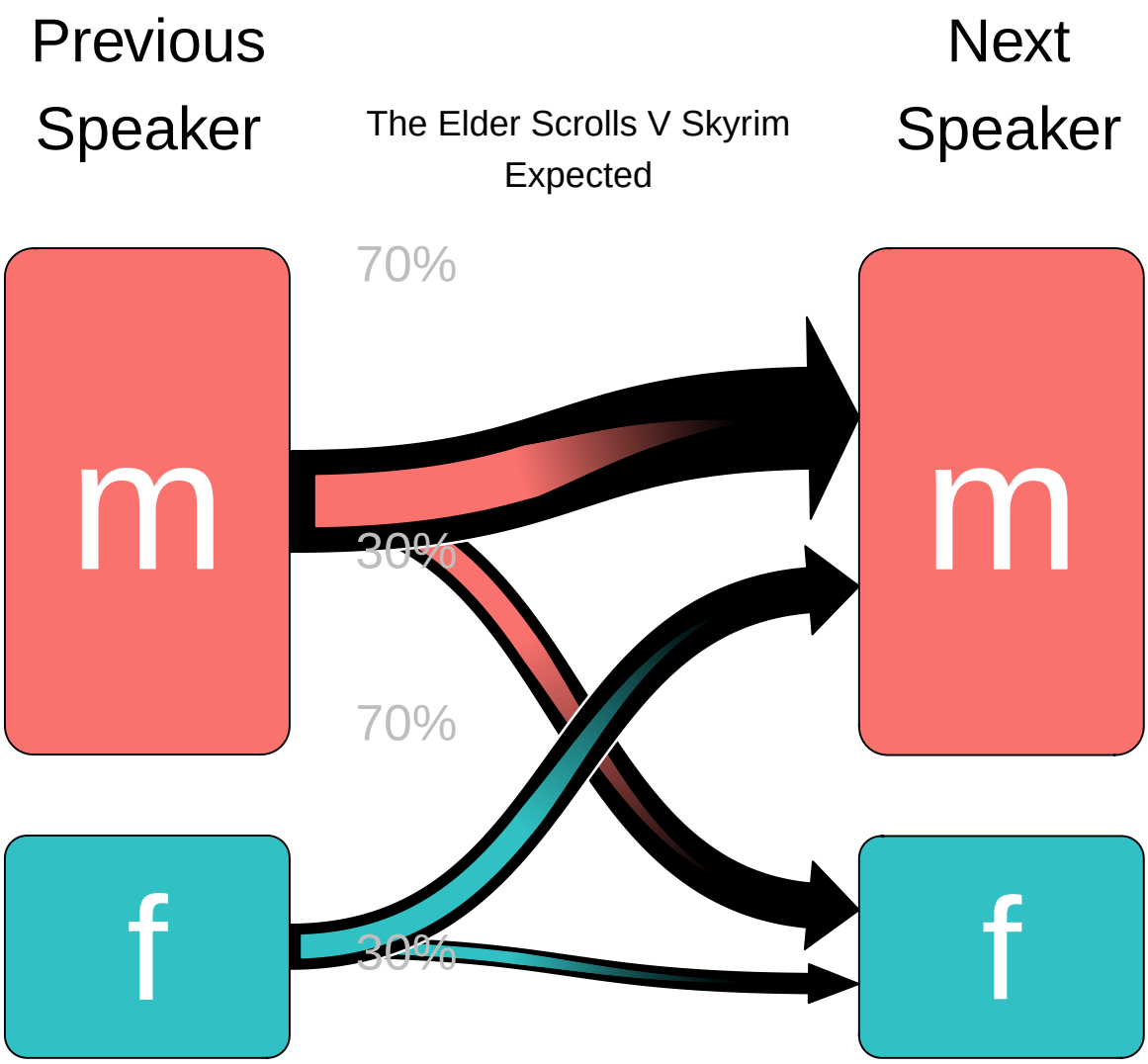

Previous Speaker
Dragon Age 2 Empirical
Next Speaker
m
f
54%
46%
63%
37%
m
f

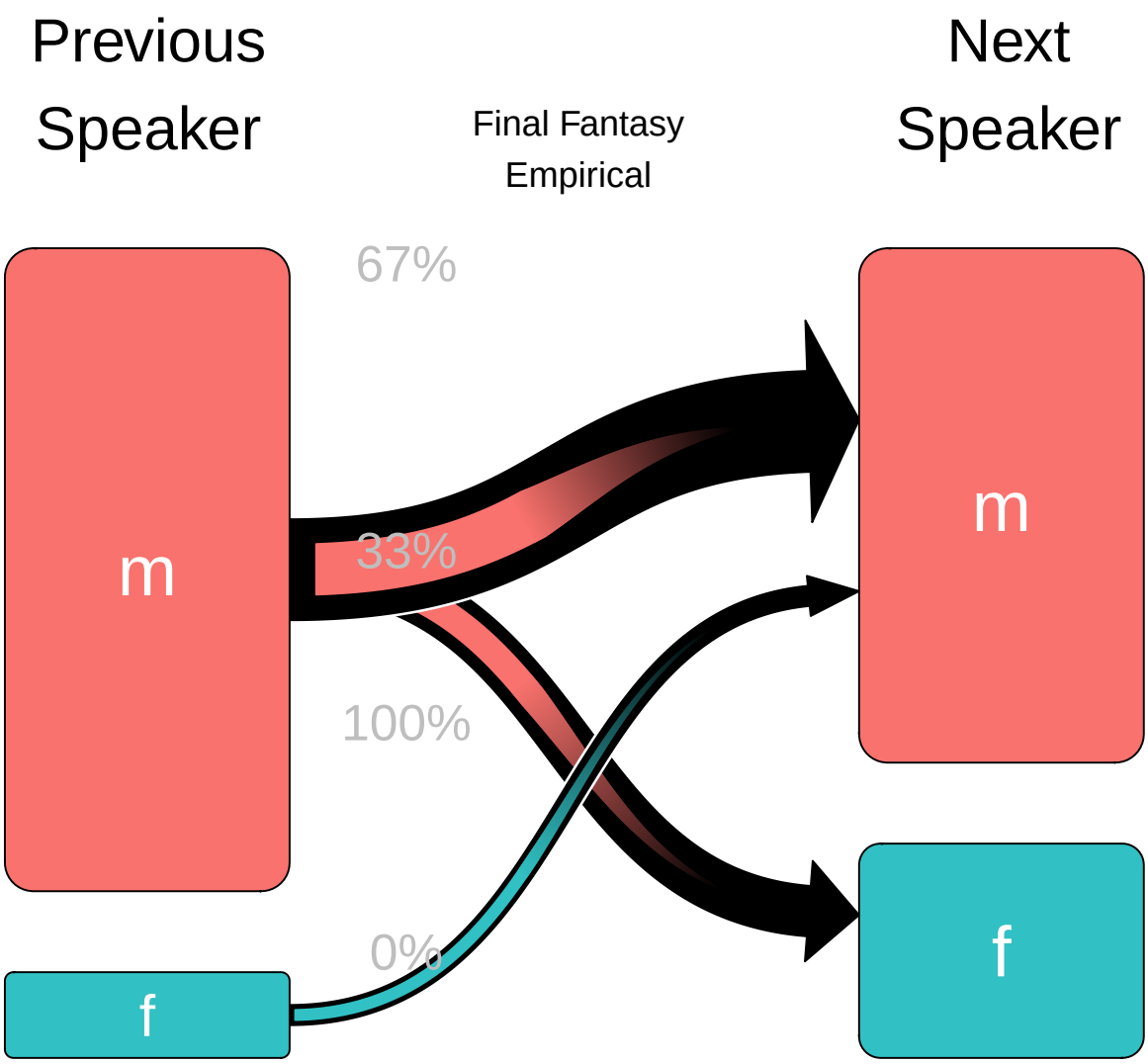

Previous Speaker
Dragon Age 2 Expected
Next Speaker
m
f
59%
41%
59%
41%
m
f

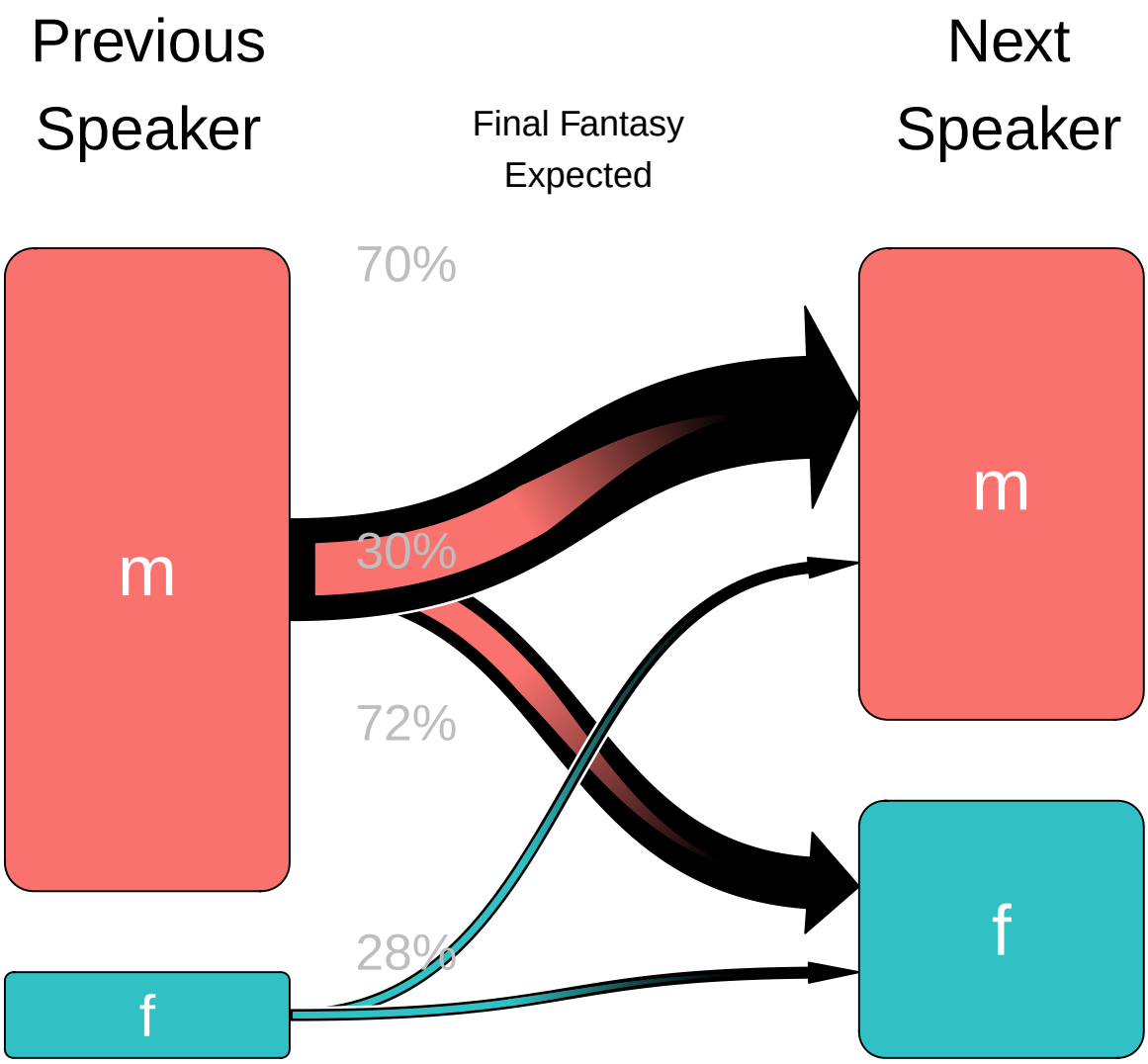

Previous Speaker
Dragon Age Origins Empirical
Next Speaker
m
f
64%
36%
63%
37%
m
f

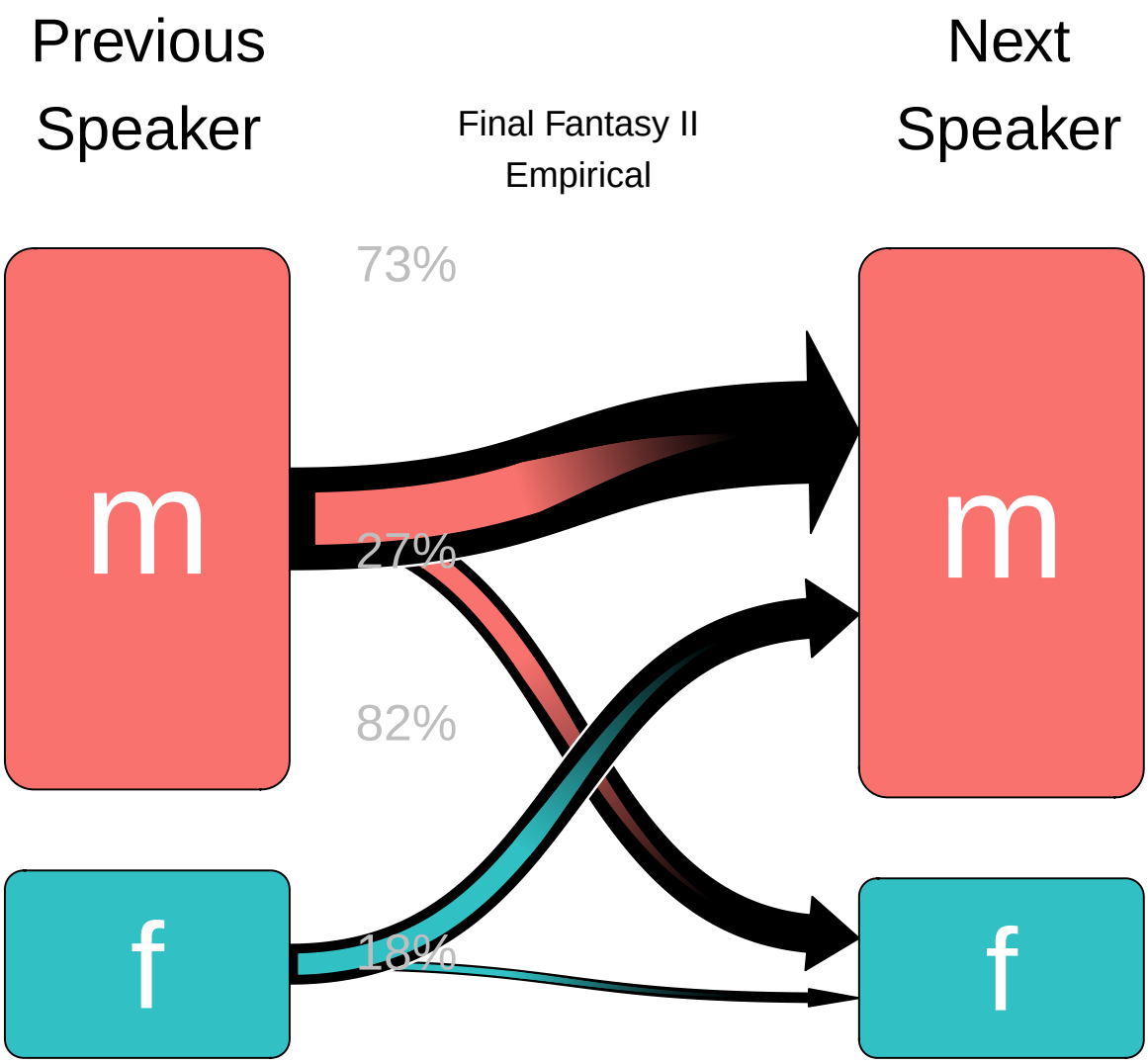

Previous Speaker
Dragon Age Origins Expected
Next Speaker
m
f
69%
31%
69%
31%
m
f

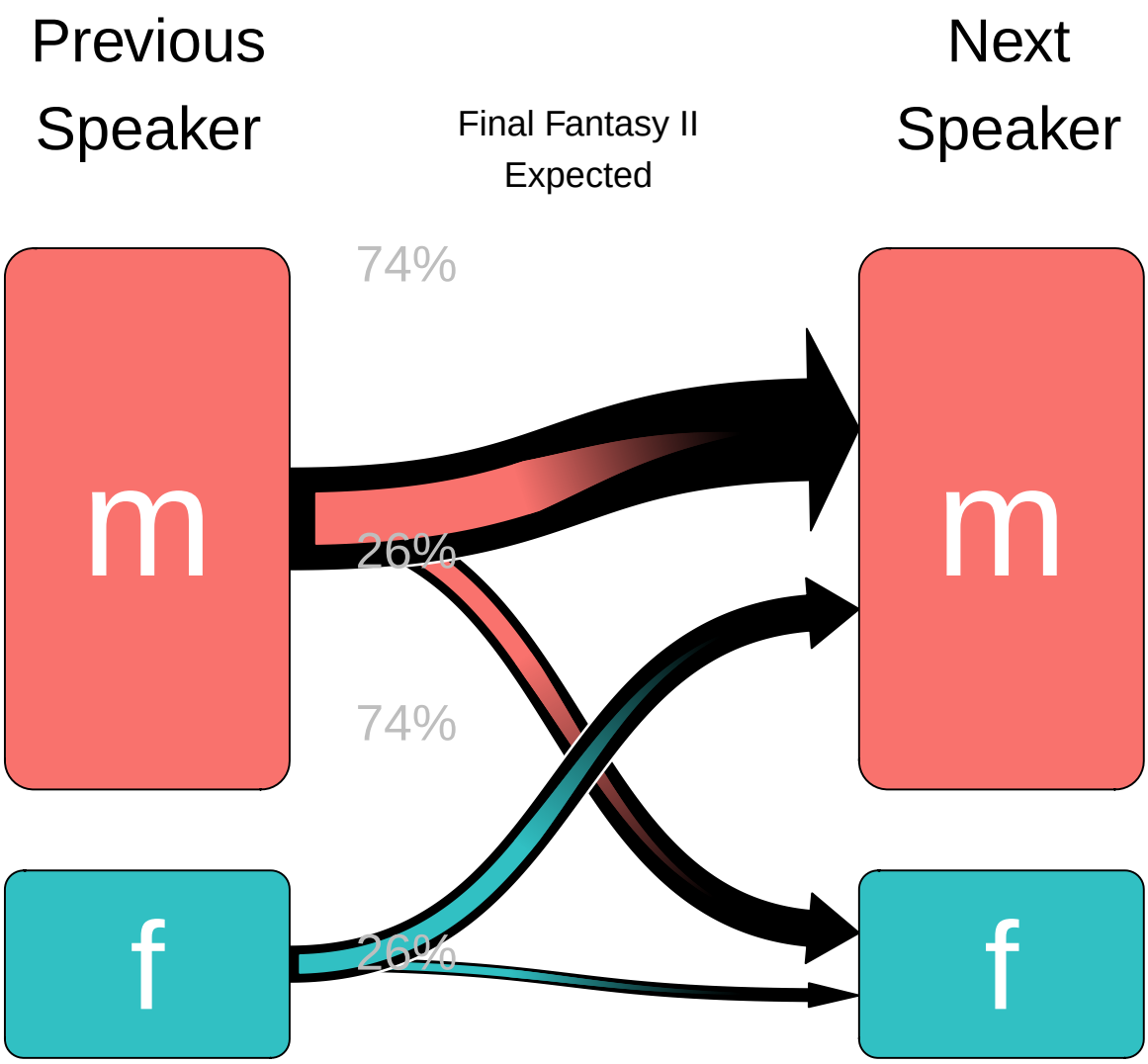

Previous Speaker
The Elder Scrolls V Skyrim Empirical
Next Speaker
m
f
65%
35%
85%
15%
m
f

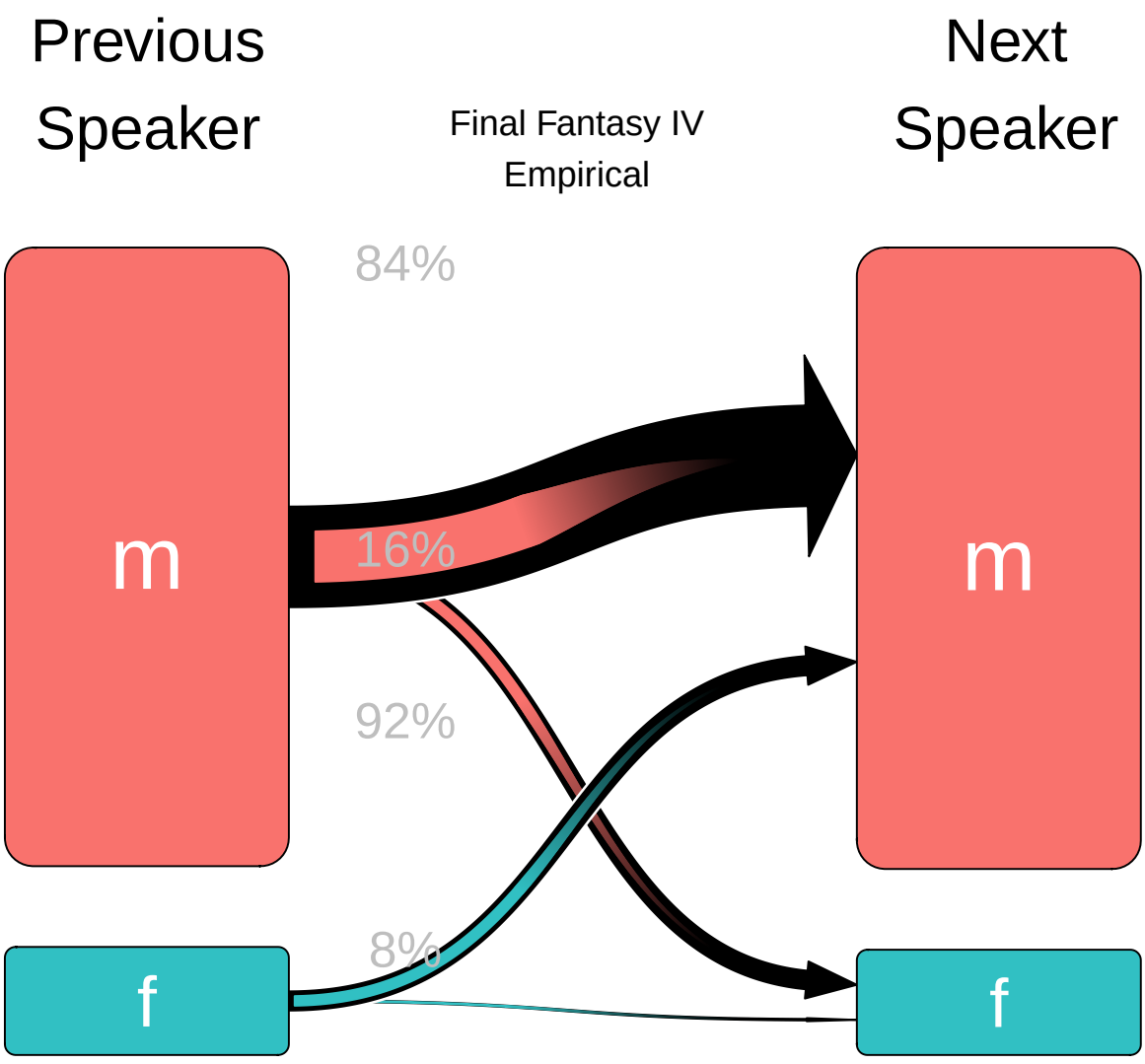

Previous Speaker
The Elder Scrolls V Skyrim
Expected
Next Speaker
m
f
70%
30%
70%
30%
m
f

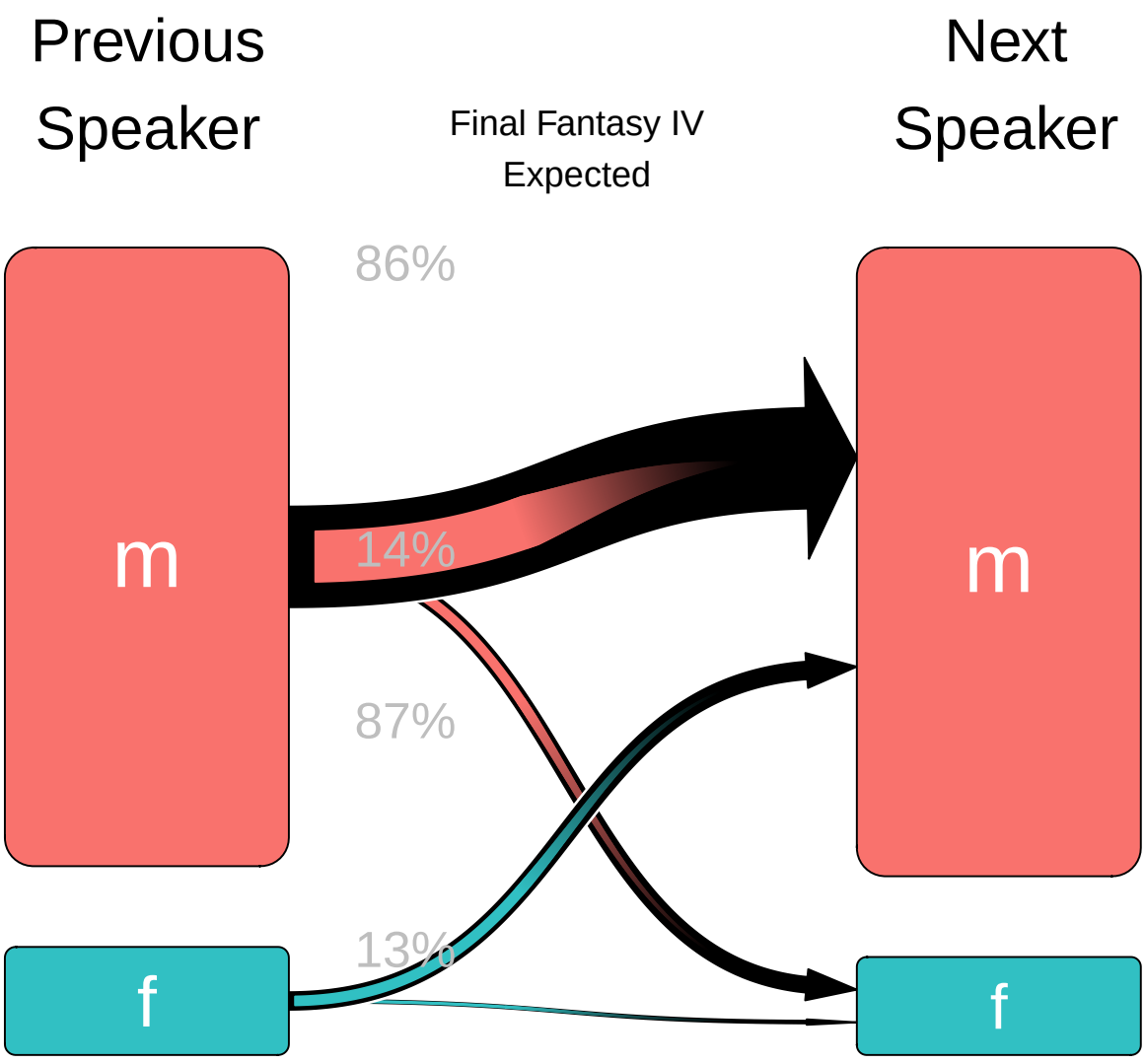

Previous Speaker
Final Fantasy Empirical
Next Speaker
m
f
67%
33%
100%
0%
m
f

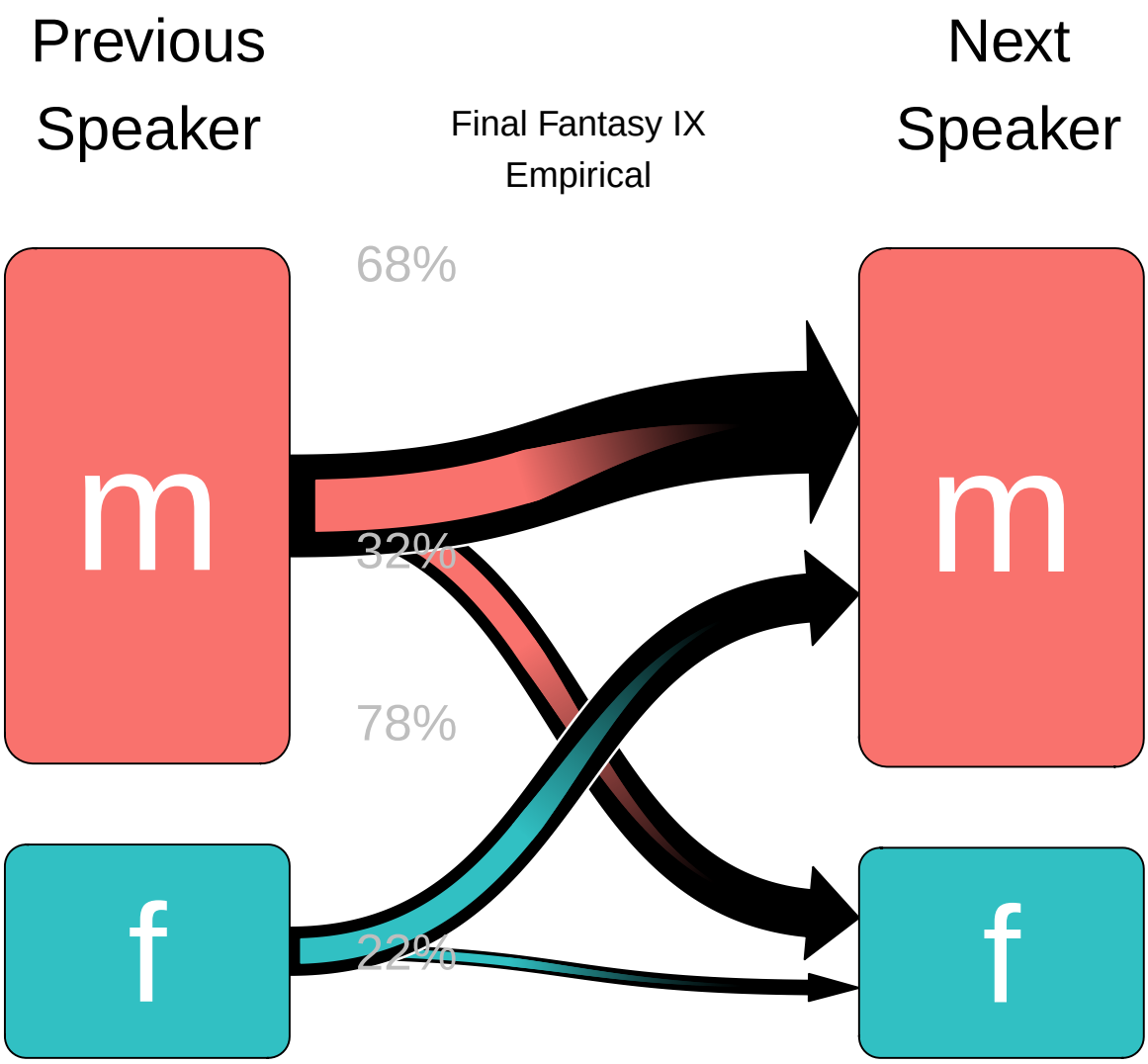

Previous Speaker
Final Fantasy Expected
Next Speaker
m
f
m
f
70%
30%
72%
28%

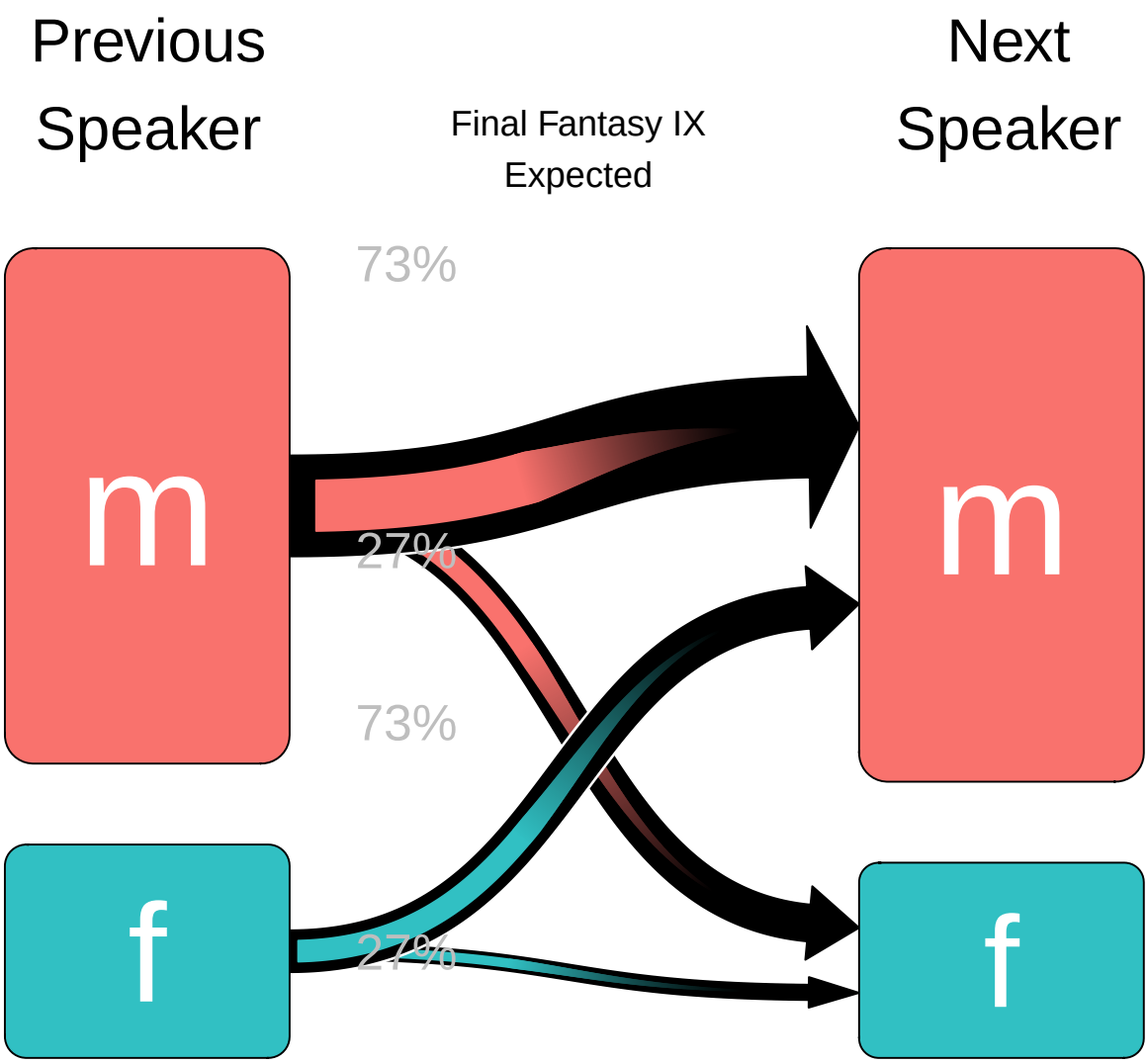

Previous Speaker
Final Fantasy II
Empirical
Next Speaker
m
f
73%
27%
82%
18%
m
f

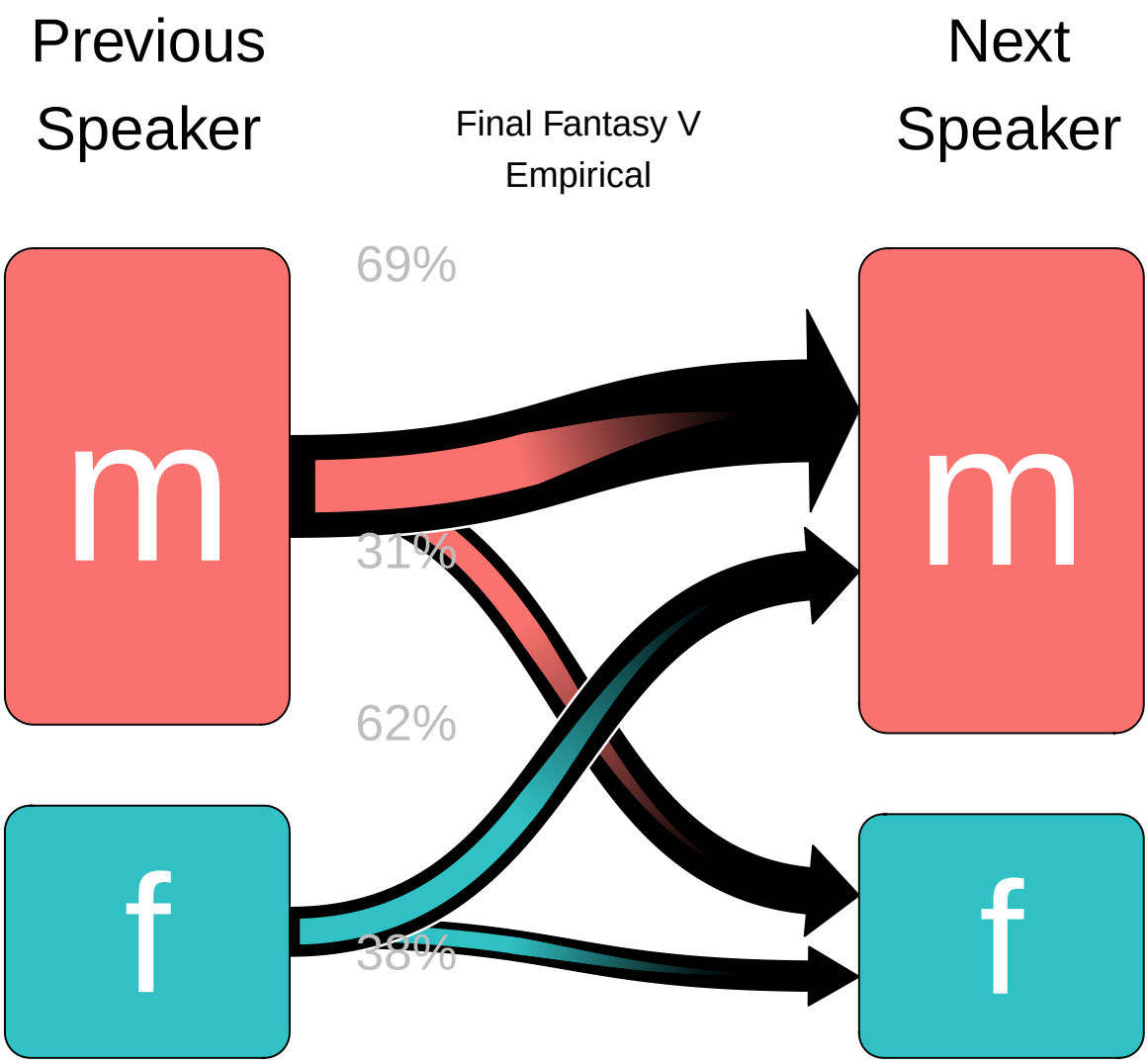

Previous Speaker
Final Fantasy II
Expected
Next Speaker
m
f
74%
26%
74%
26%
m
f

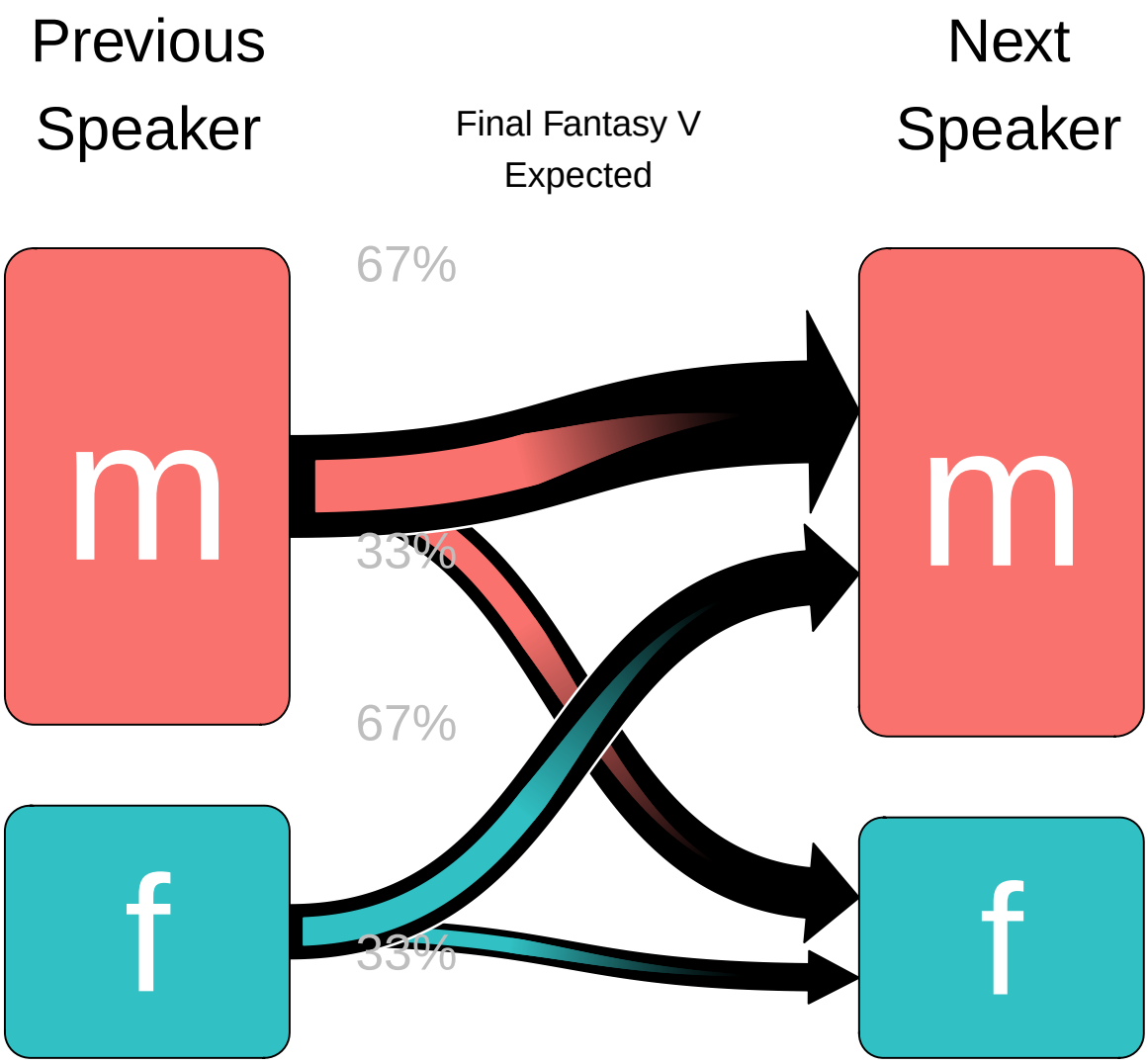

Previous Speaker
Final Fantasy IV
Empirical
Next Speaker
84%
16%
92%
8%
m
f
m
f

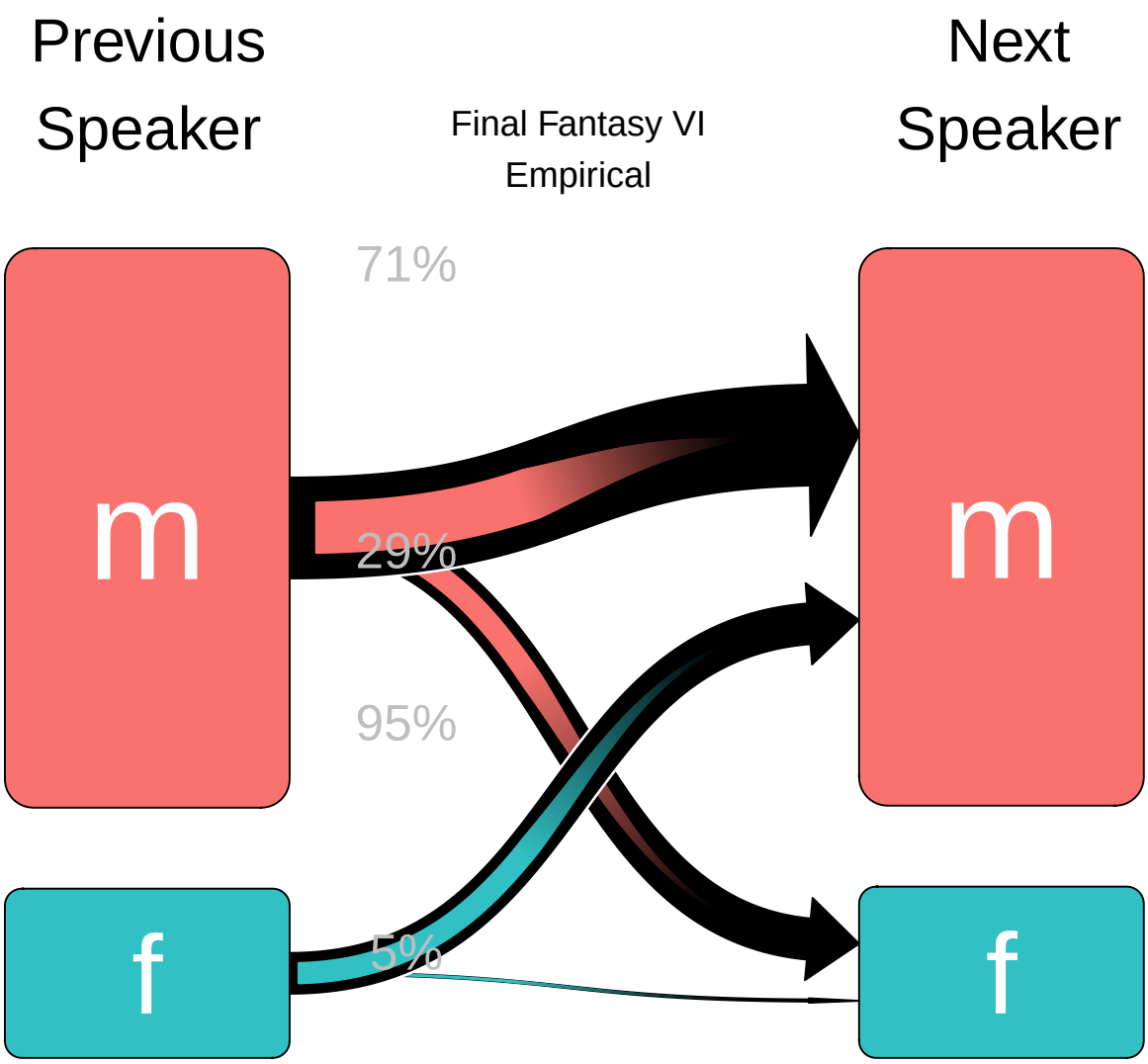

Previous Speaker
Final Fantasy IV Expected
Next Speaker
m
f
86%
14%
87%
13%
m
f

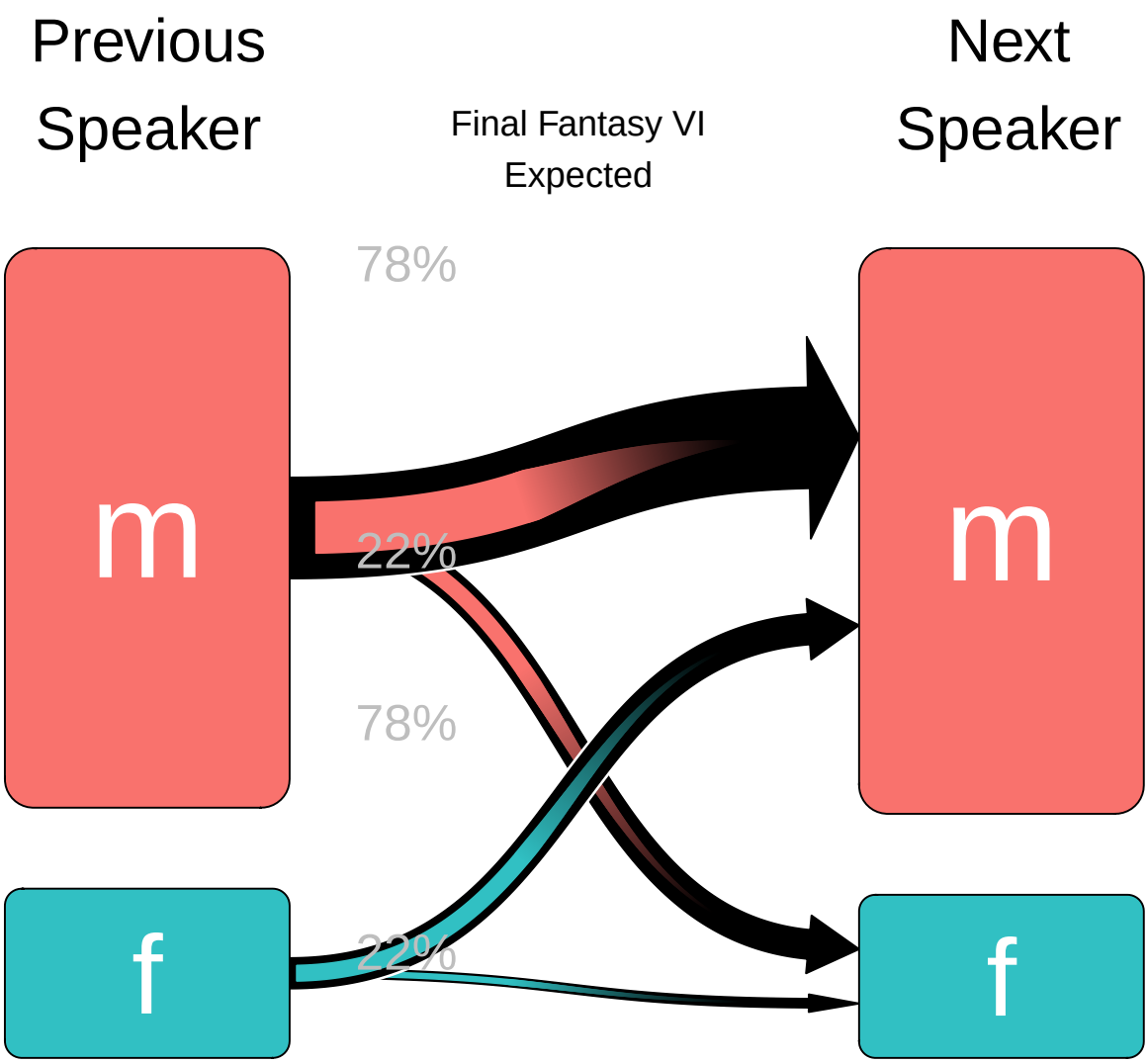

Previous Speaker
Final Fantasy IX Empirical
Next Speaker
m
f
68%
32%
78%
22%
m
f

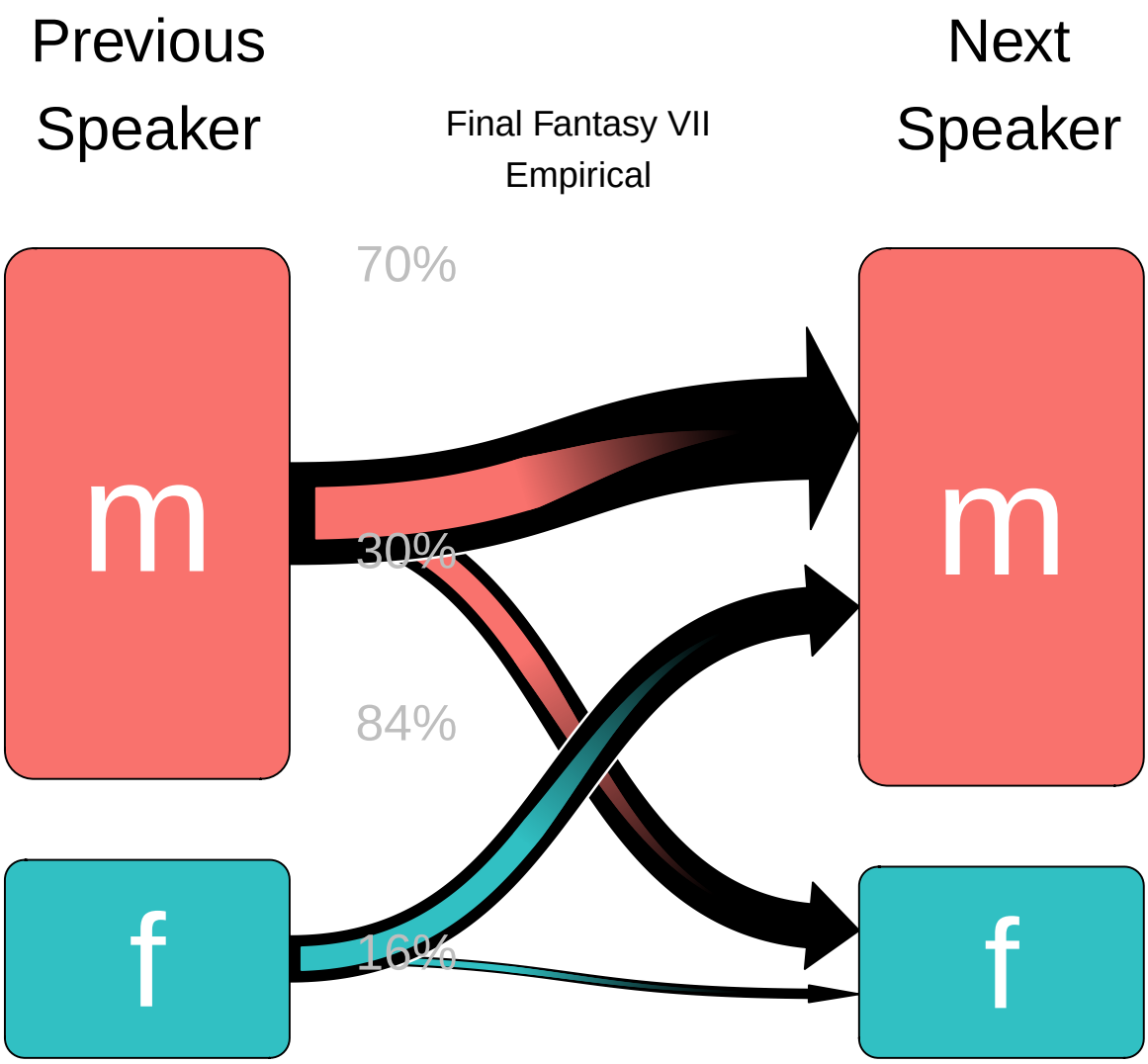

Previous Speaker
Final Fantasy IX
Expected
Next Speaker
m
f
73%
27%
73%
27%
m
f

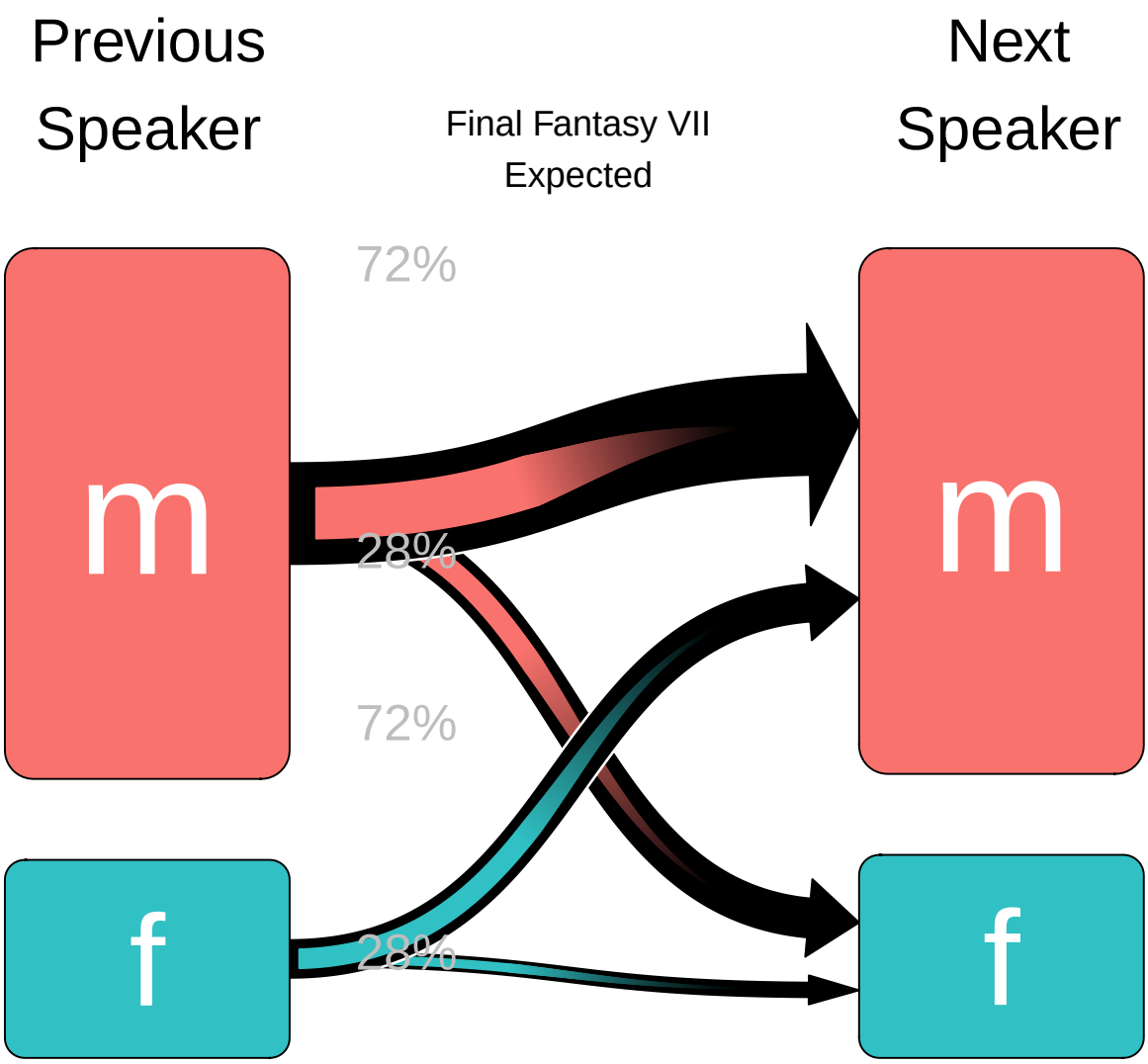

Previous Speaker
Final Fantasy V
Empirical
Next Speaker
m
f
69%
31%
62%
38%
m
f

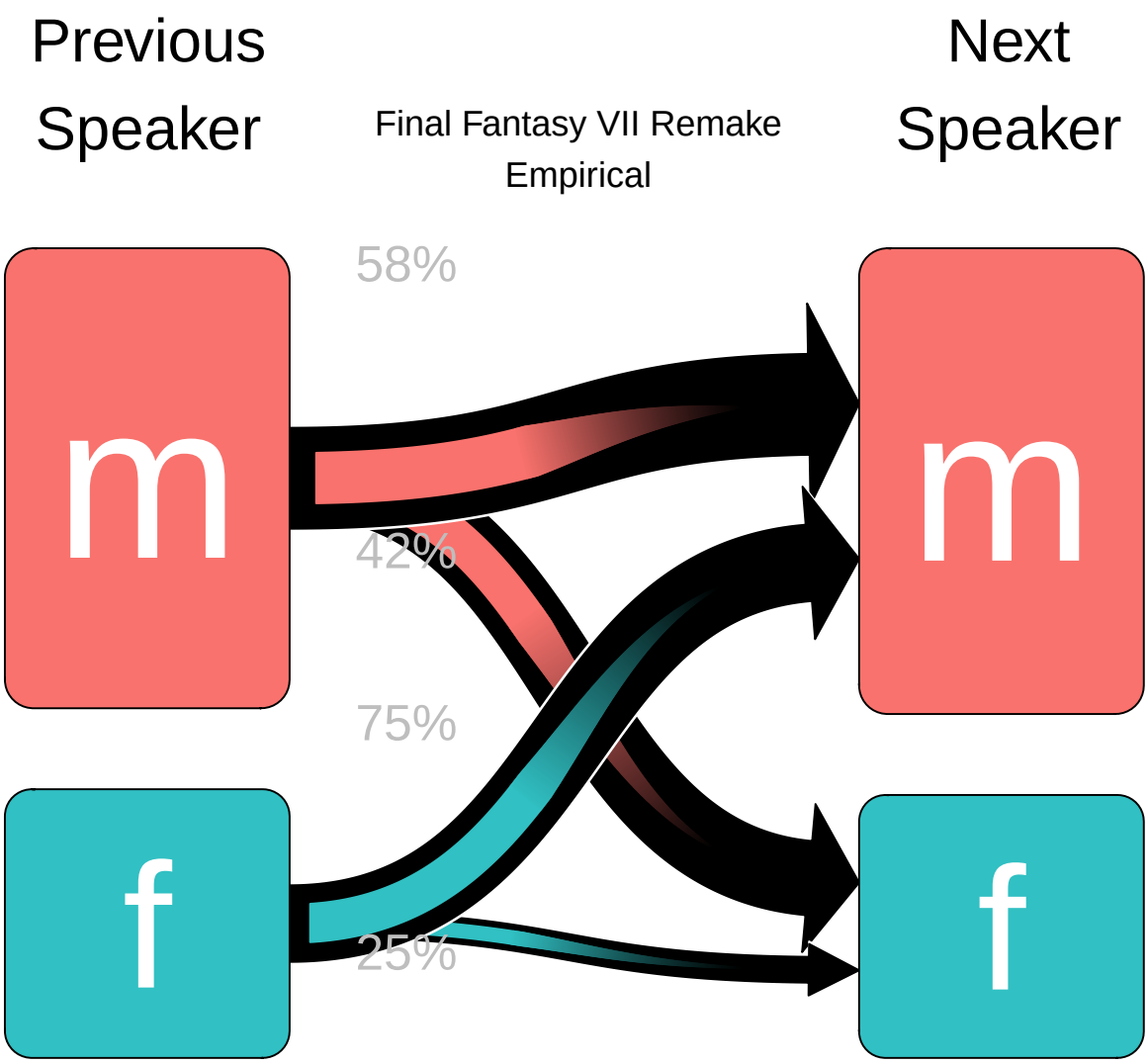

Previous Speaker
Final Fantasy V Expected
Next Speaker
m
f
67%
33%
67%
33%
m
f

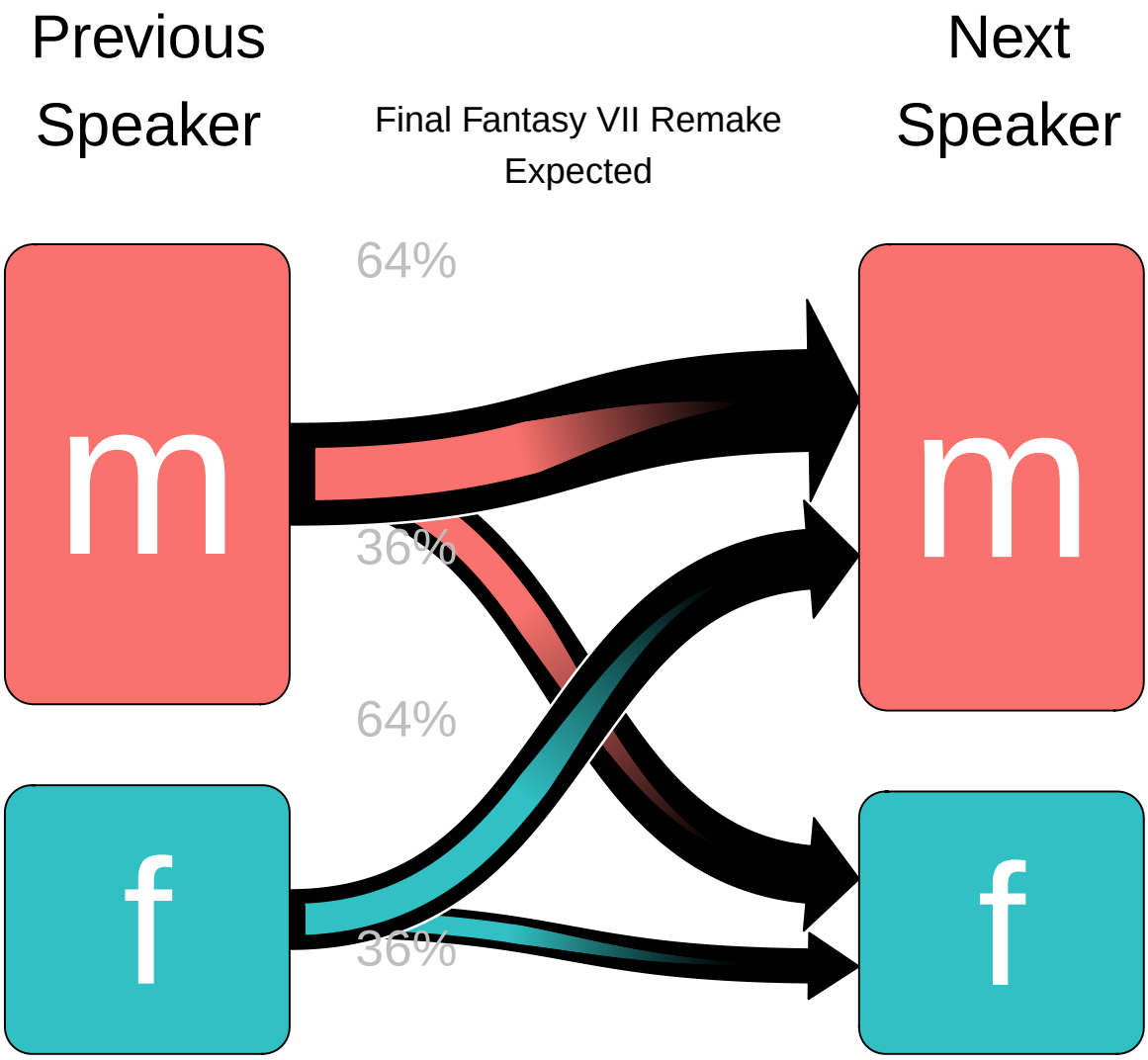

Previous Speaker
Final Fantasy VI Empirical
Next Speaker
m
f
71%
29%
95%
5%
m
f

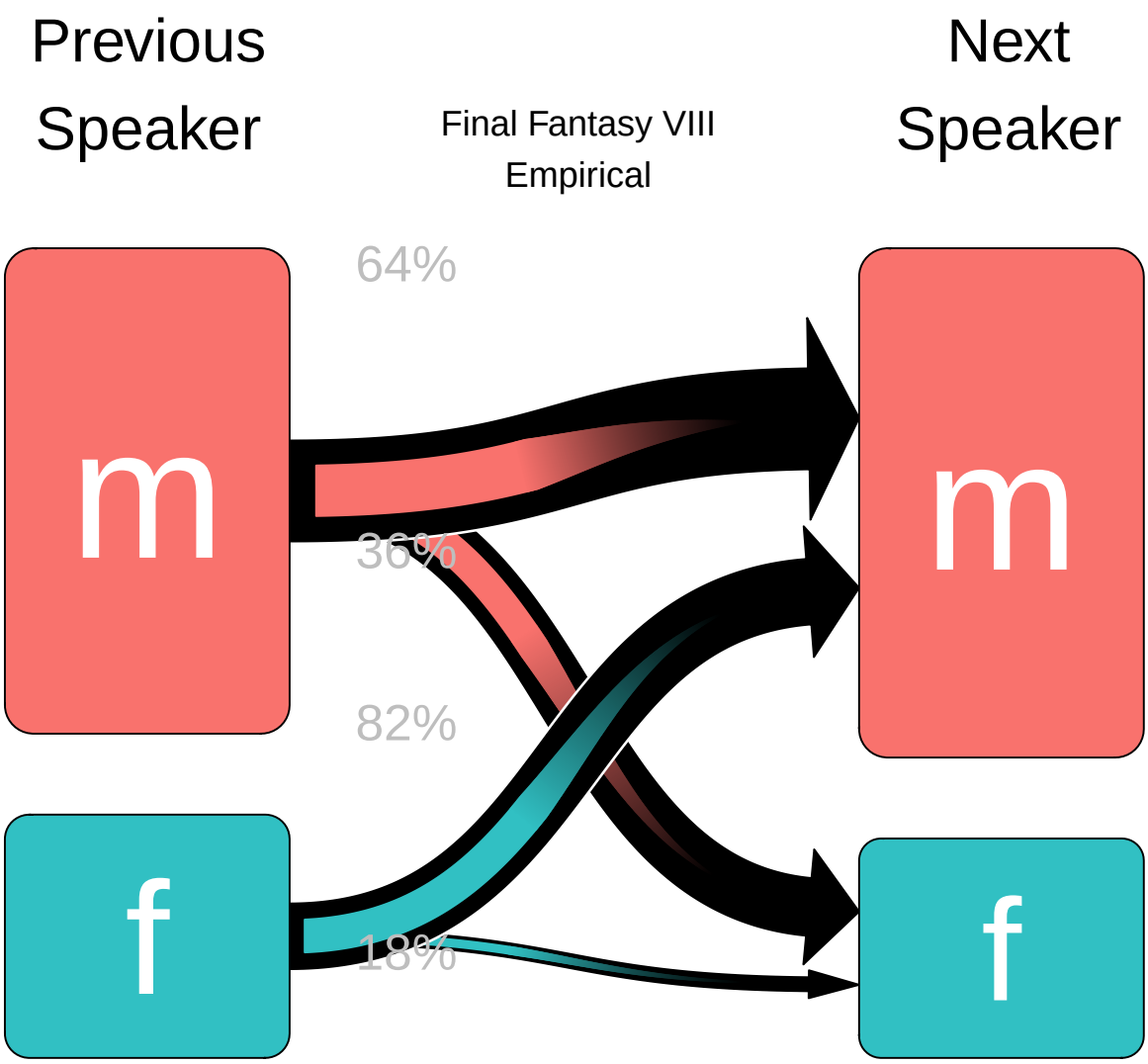

Previous Speaker
Final Fantasy VI Expected
Next Speaker
m
f
78%
22%
78%
22%
m
f

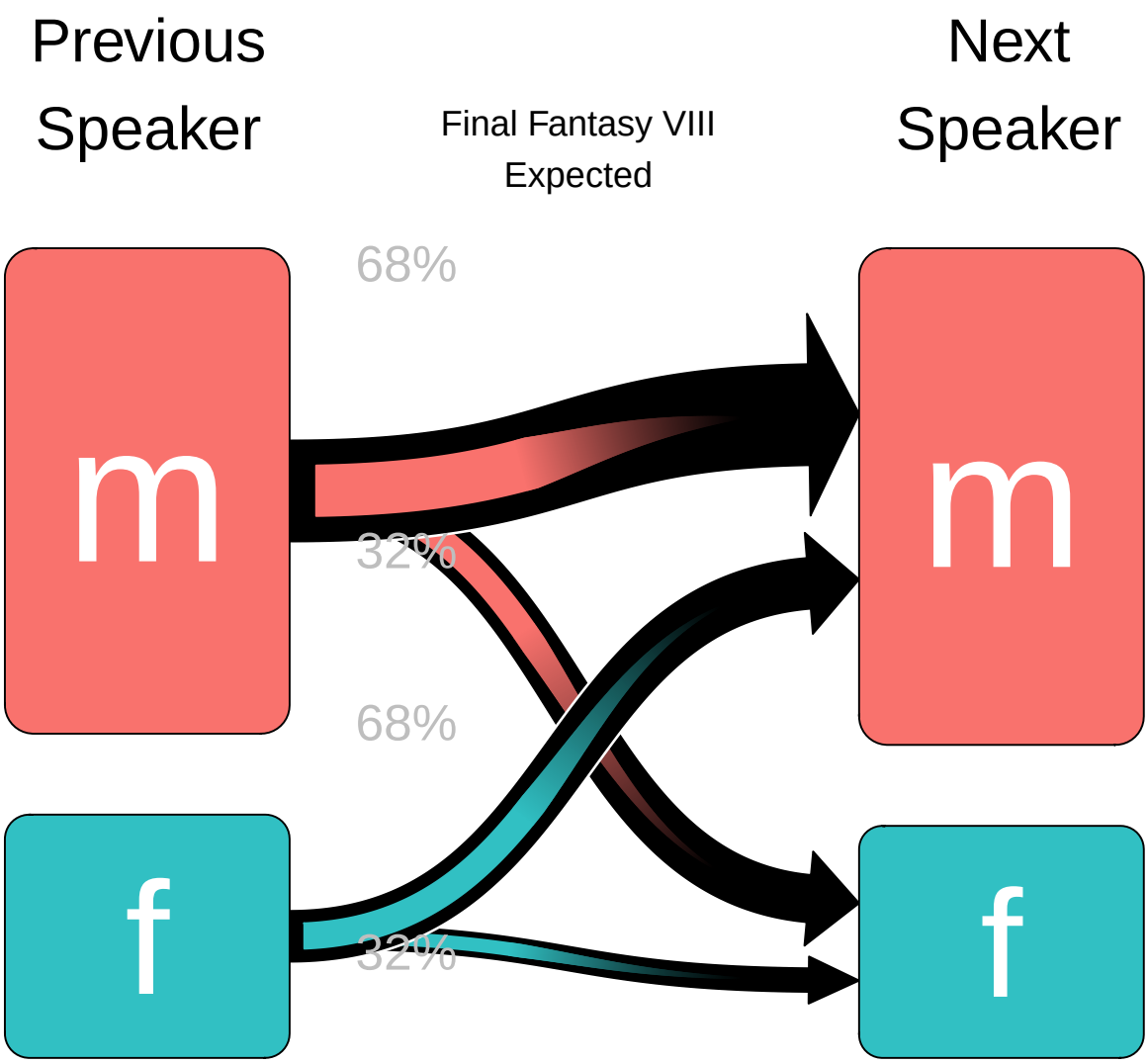

Previous Speaker
Final Fantasy VII
Empirical
Next Speaker
m
f
m
f
70%
30%
84%
16%

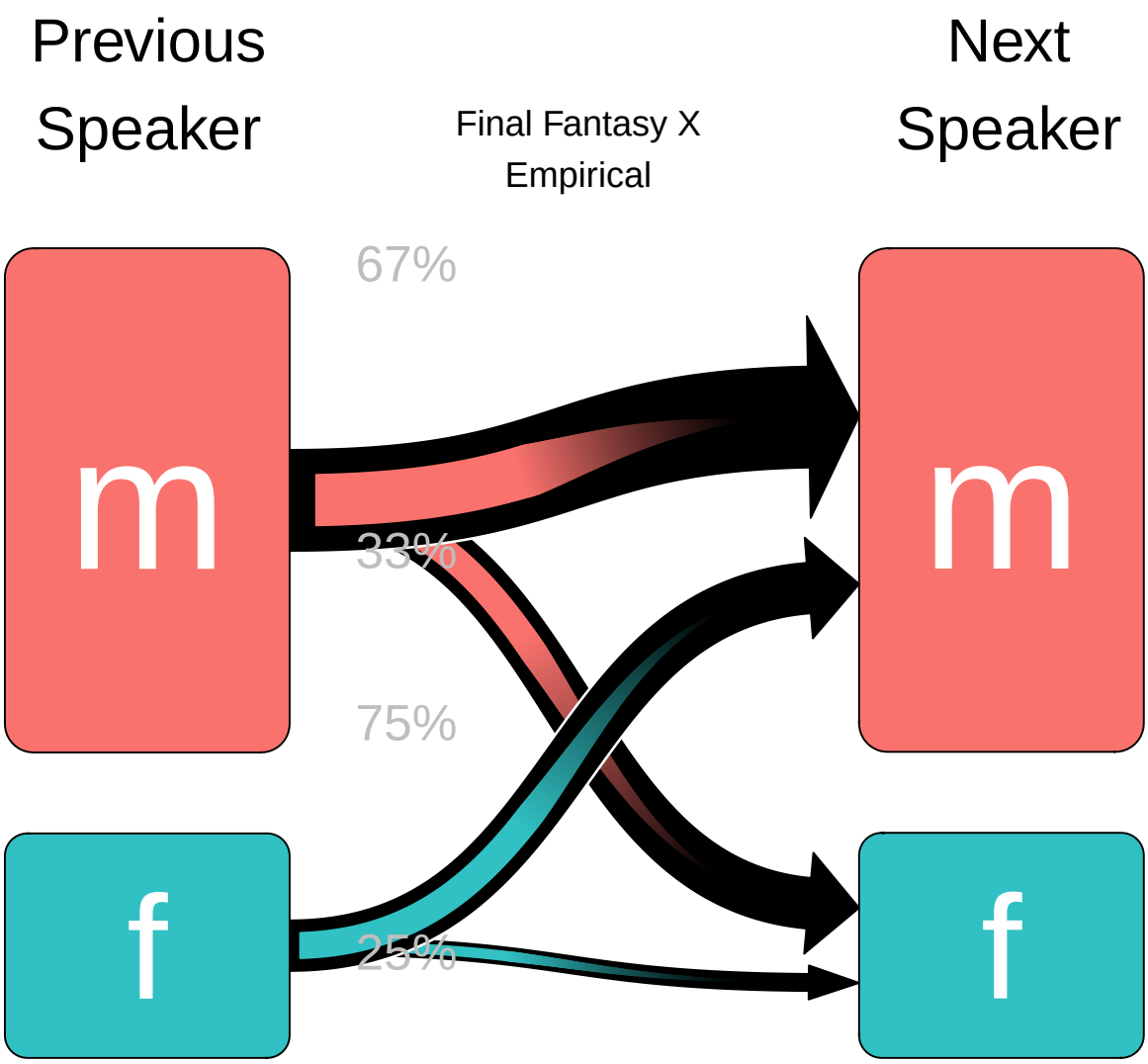

Previous Speaker
Final Fantasy VII Expected
Next Speaker
m
f
72%
28%
72%
28%
m
f

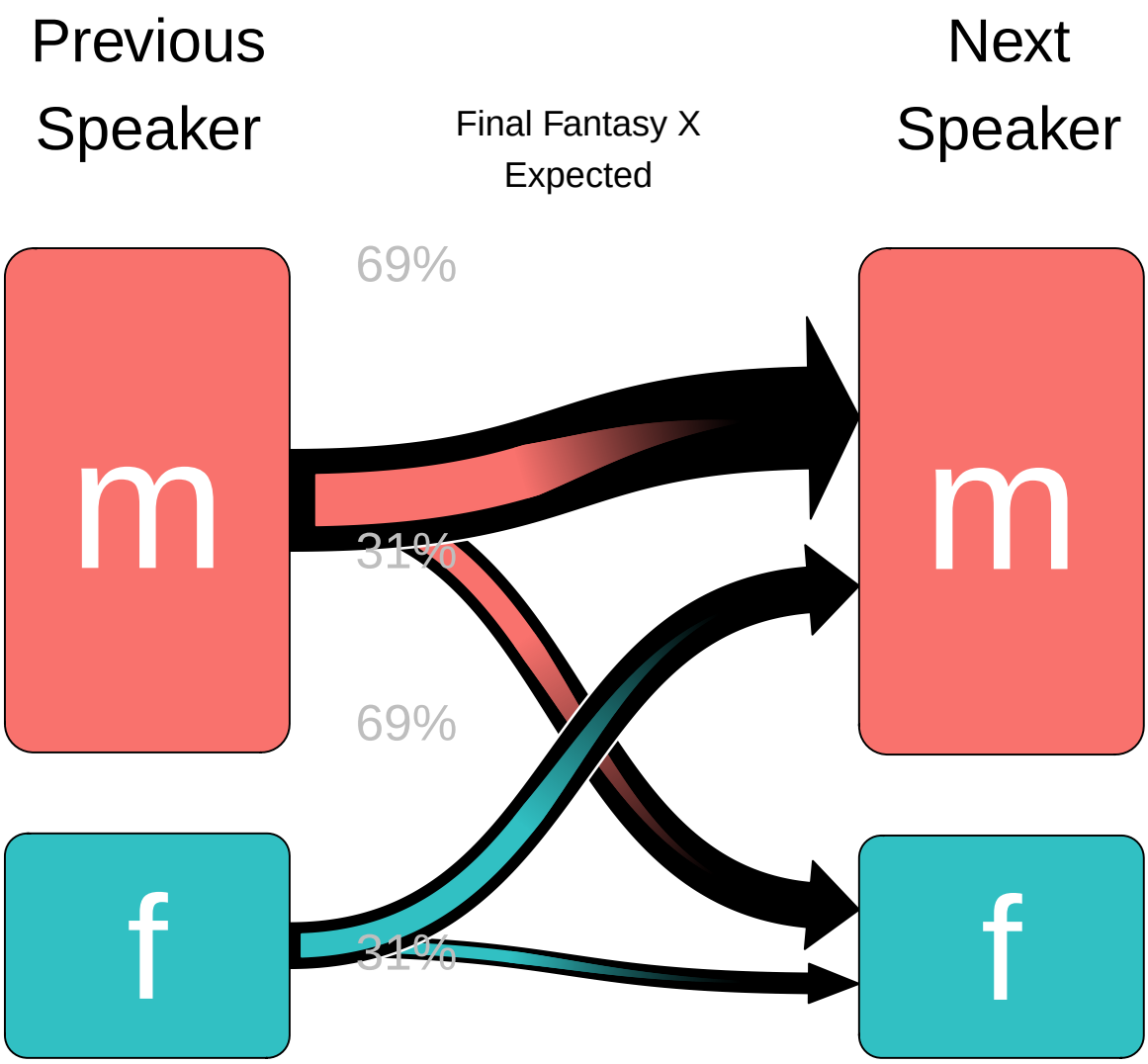

Previous Speaker
Final Fantasy VII Remake
Empirical
Next Speaker
m
f
58%
42%
75%
25%
m
f

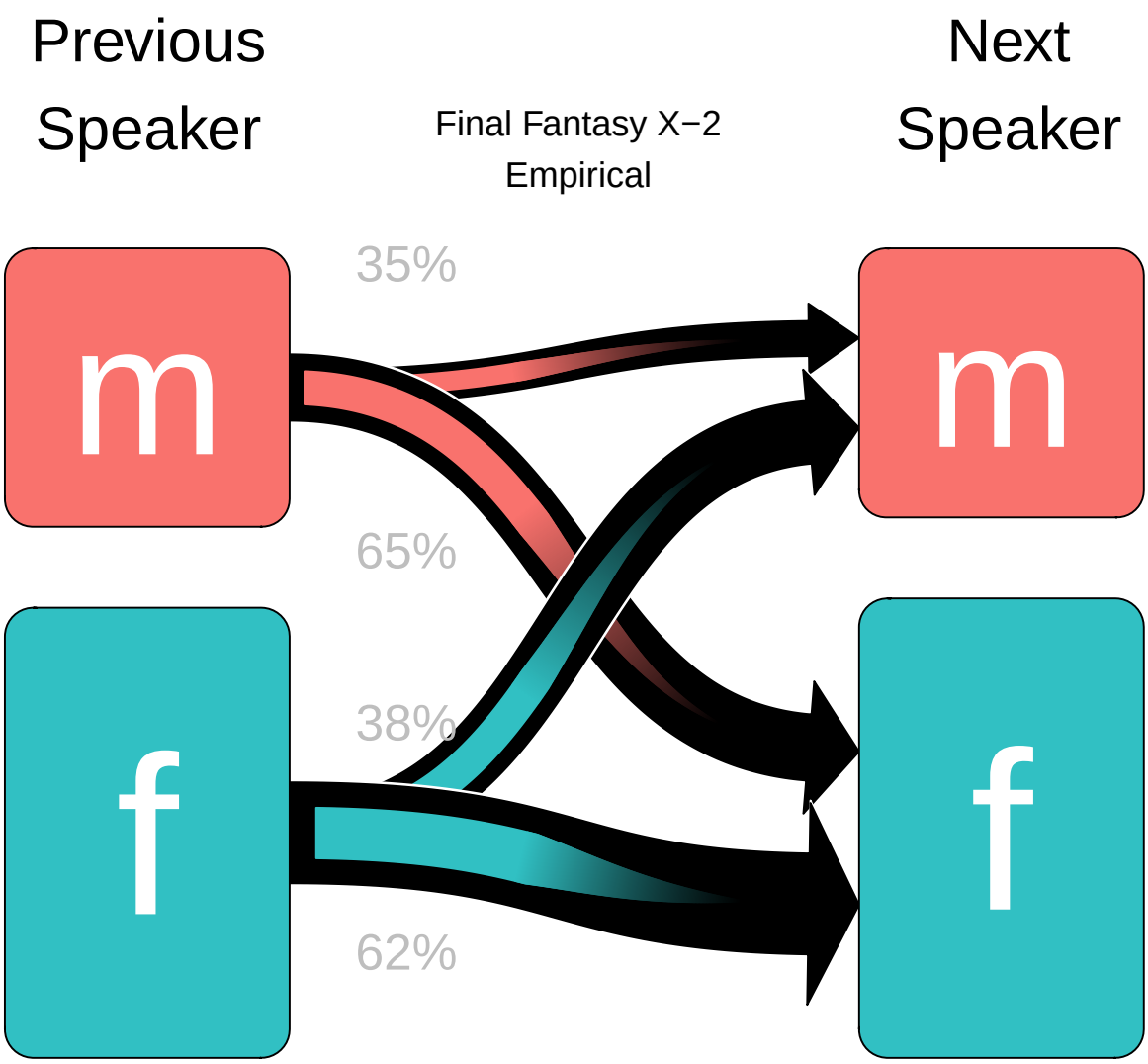

Previous Speaker
Final Fantasy VII Remake
Expected
Next Speaker
m
f
64%
36%
64%
36%
m
f

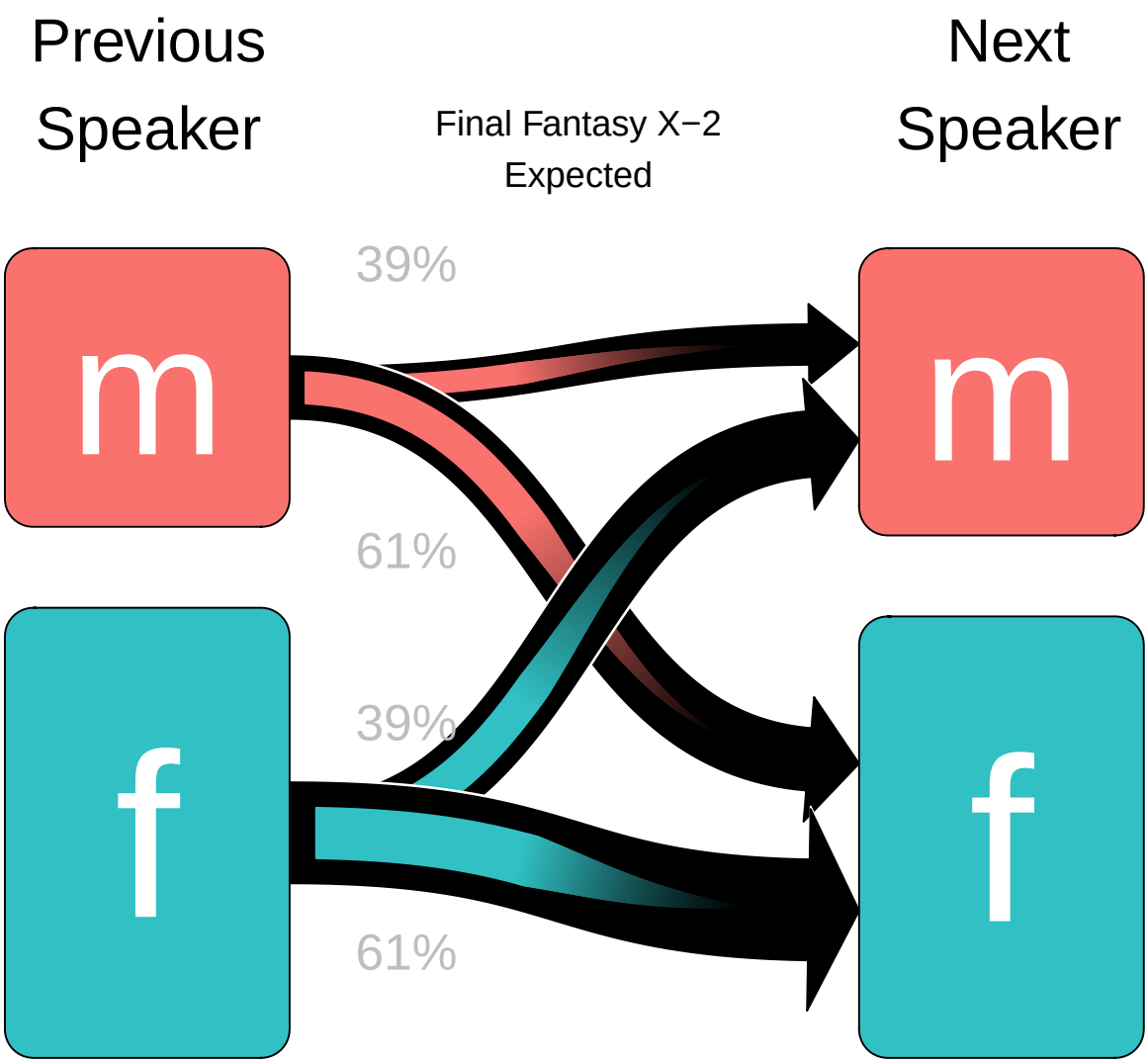

Previous Speaker
Final Fantasy VIII
Empirical
Next Speaker
m
f
64%
36%
82%
18%
m
f

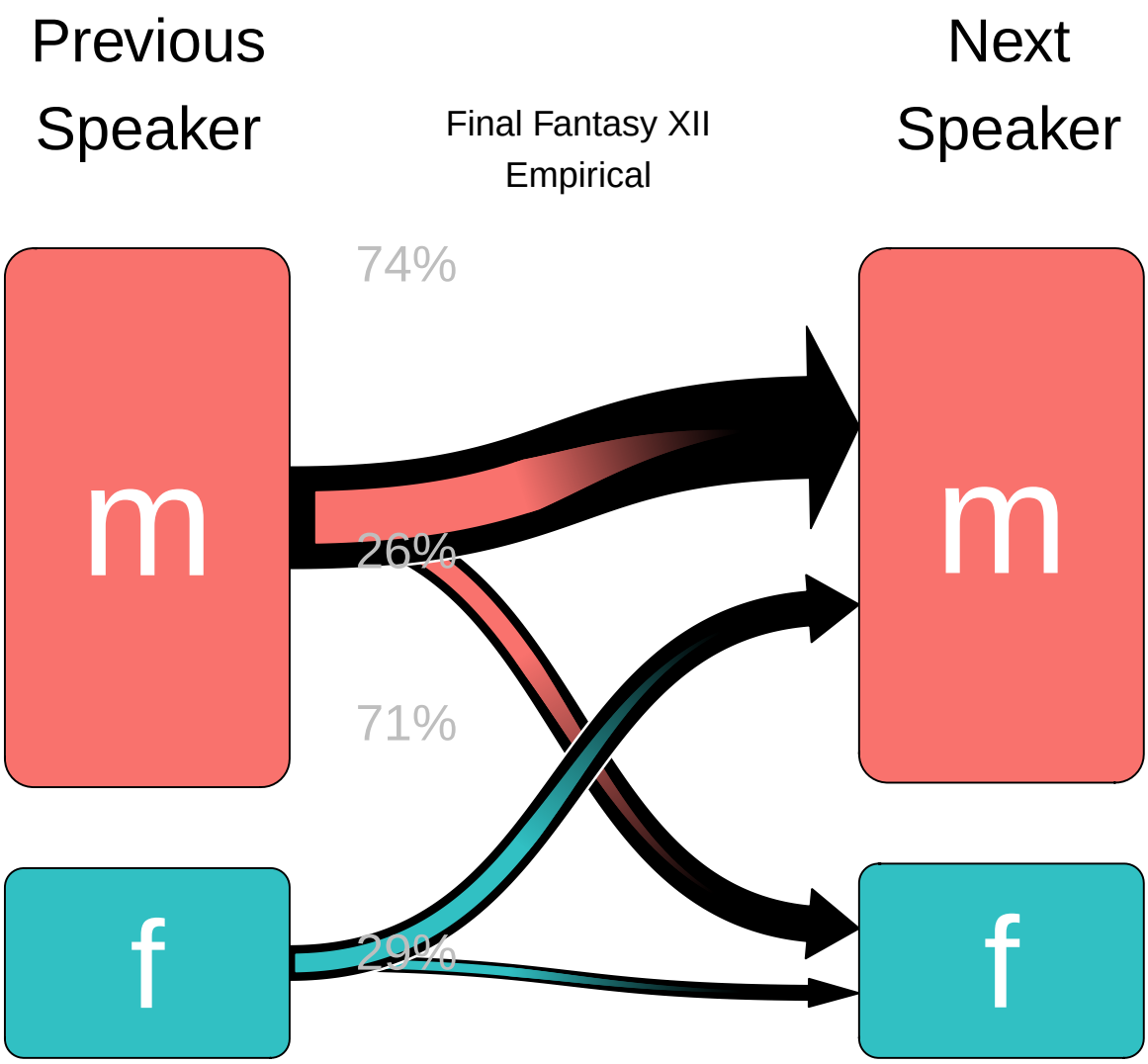

Previous Speaker
Final Fantasy VIII Expected
Next Speaker
m
f
68%
32%
68%
32%
m
f

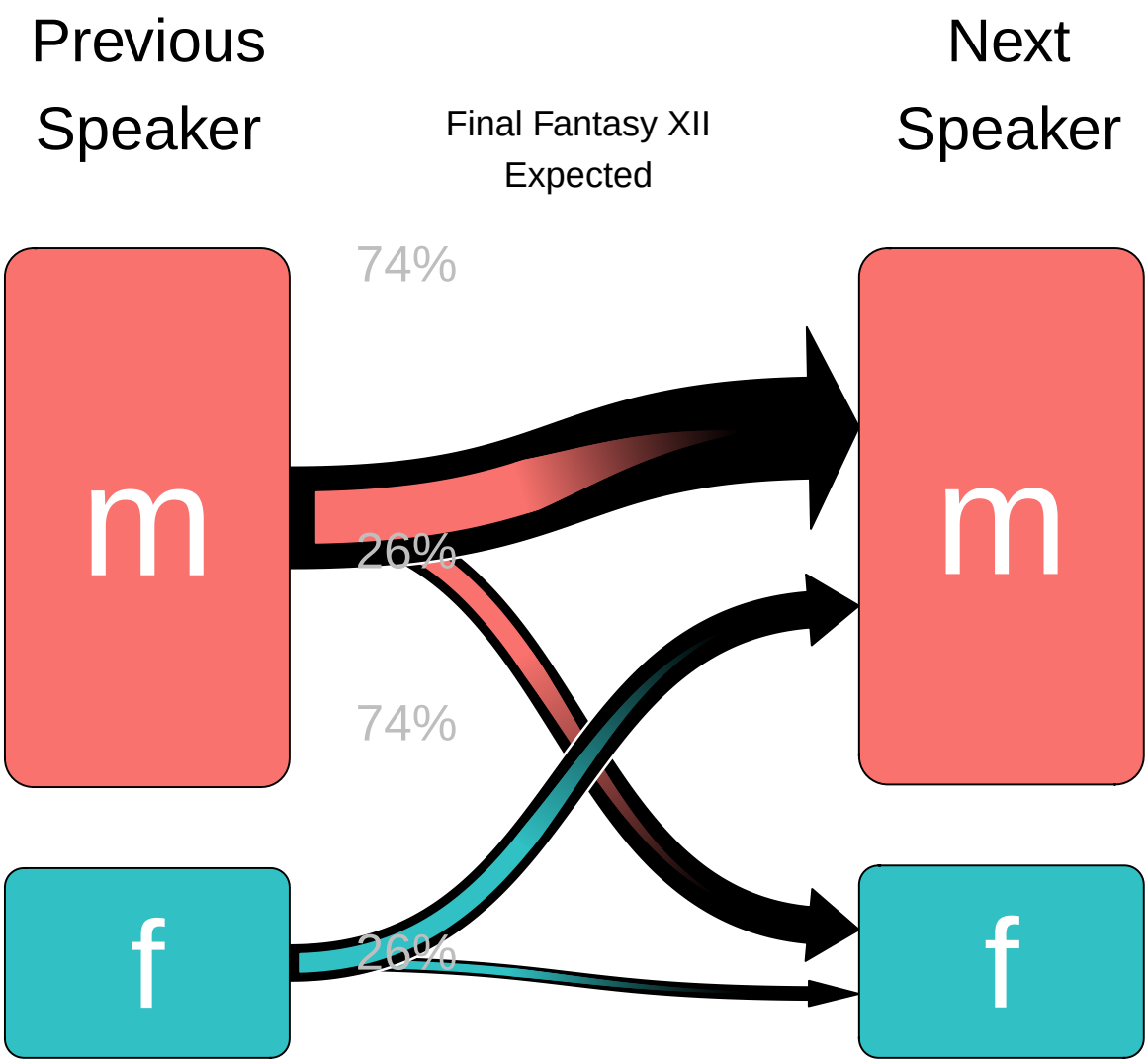

Previous Speaker
Final Fantasy X
Empirical
Next Speaker
67%
33%
75%
25%
m
f
m
f

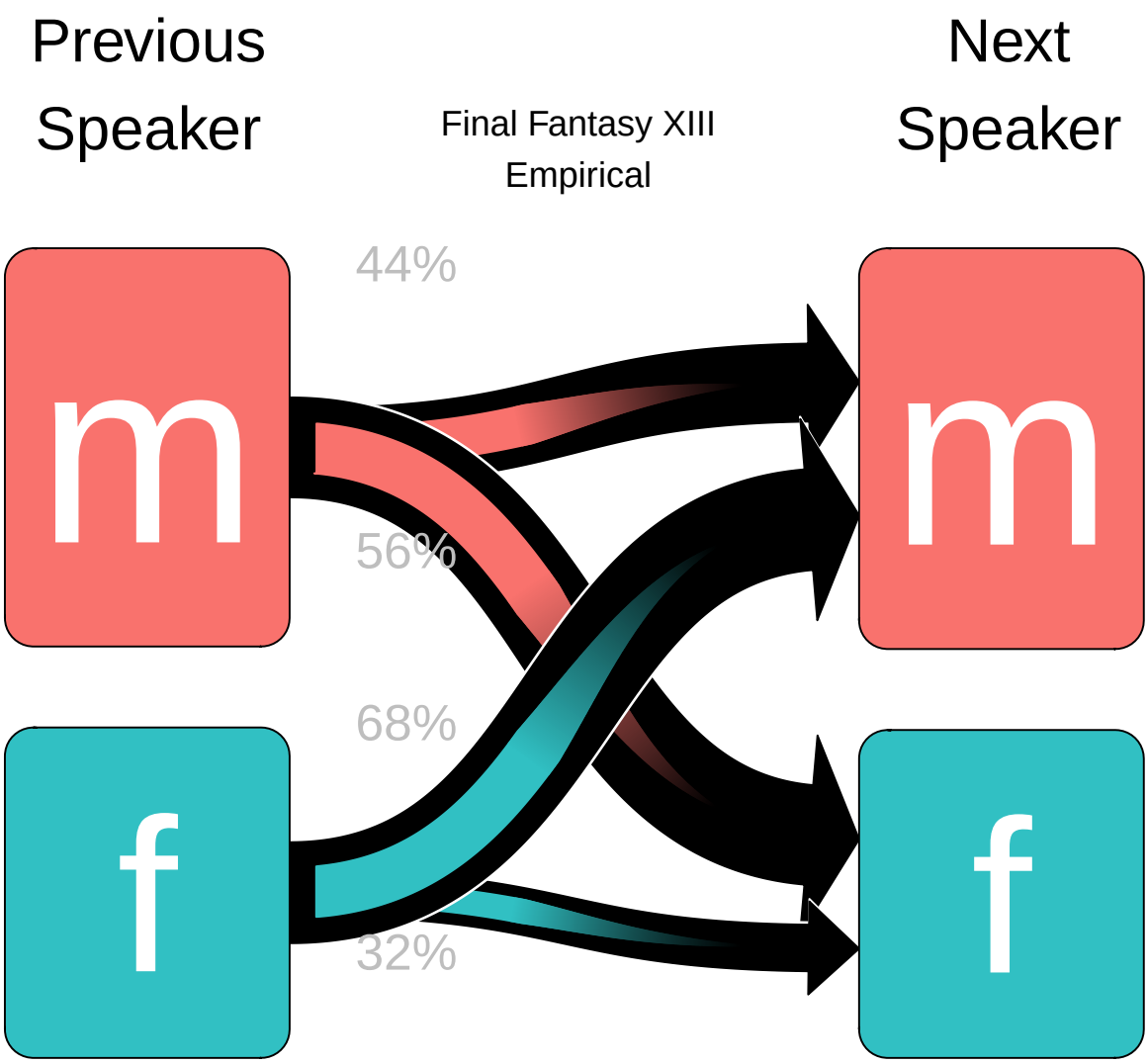

Previous Speaker
Final Fantasy X
Expected
Next Speaker
m
f
69%
31%
69%
31%
m
f

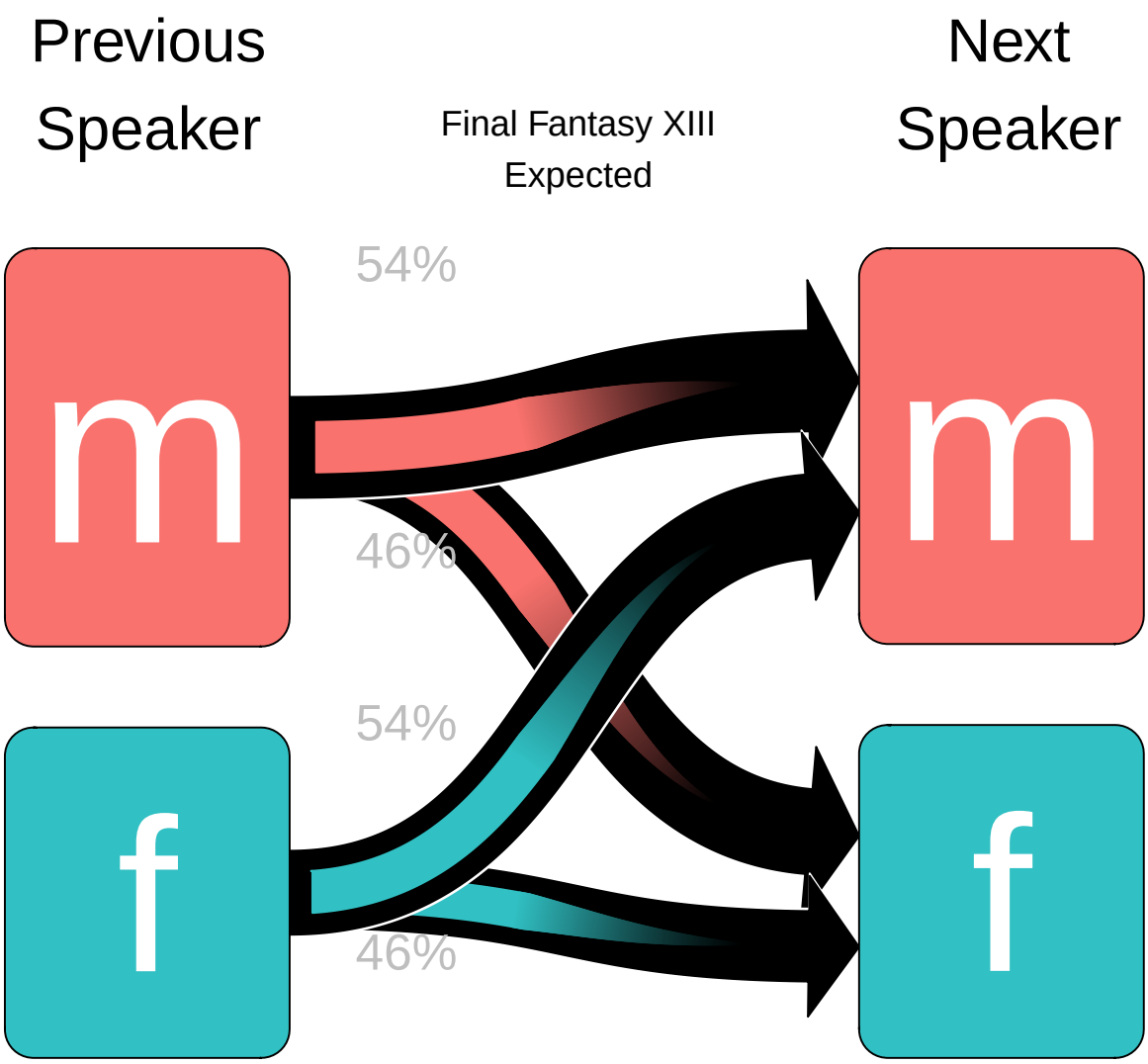

Previous Speaker
Final Fantasy X−2
Empirical
Next Speaker
m
f
35%
65%
38%
62%
m
f

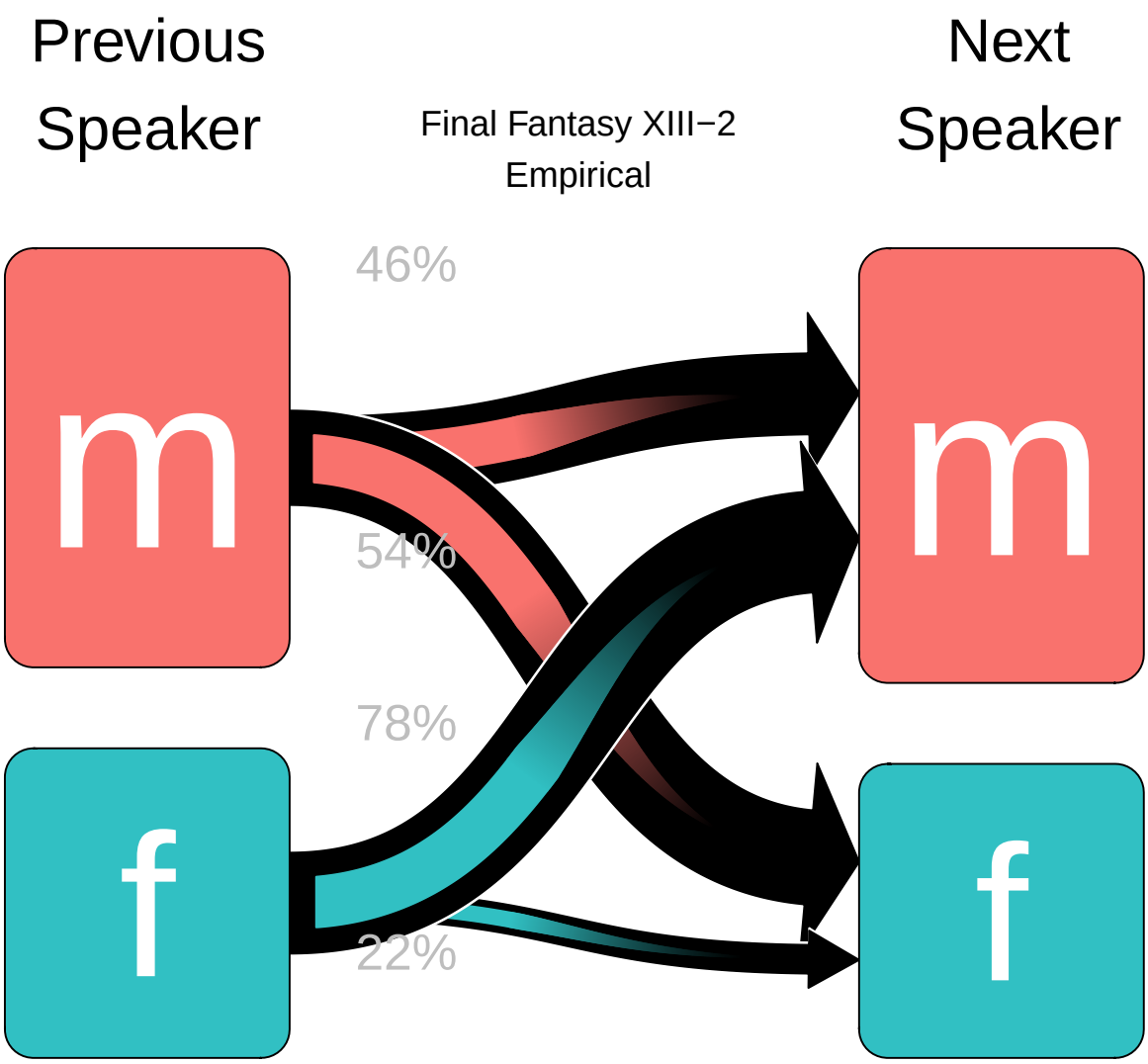

Previous Speaker
Final Fantasy X−2
Expected
Next Speaker
m
f
39%
61%
39%
61%
m
f

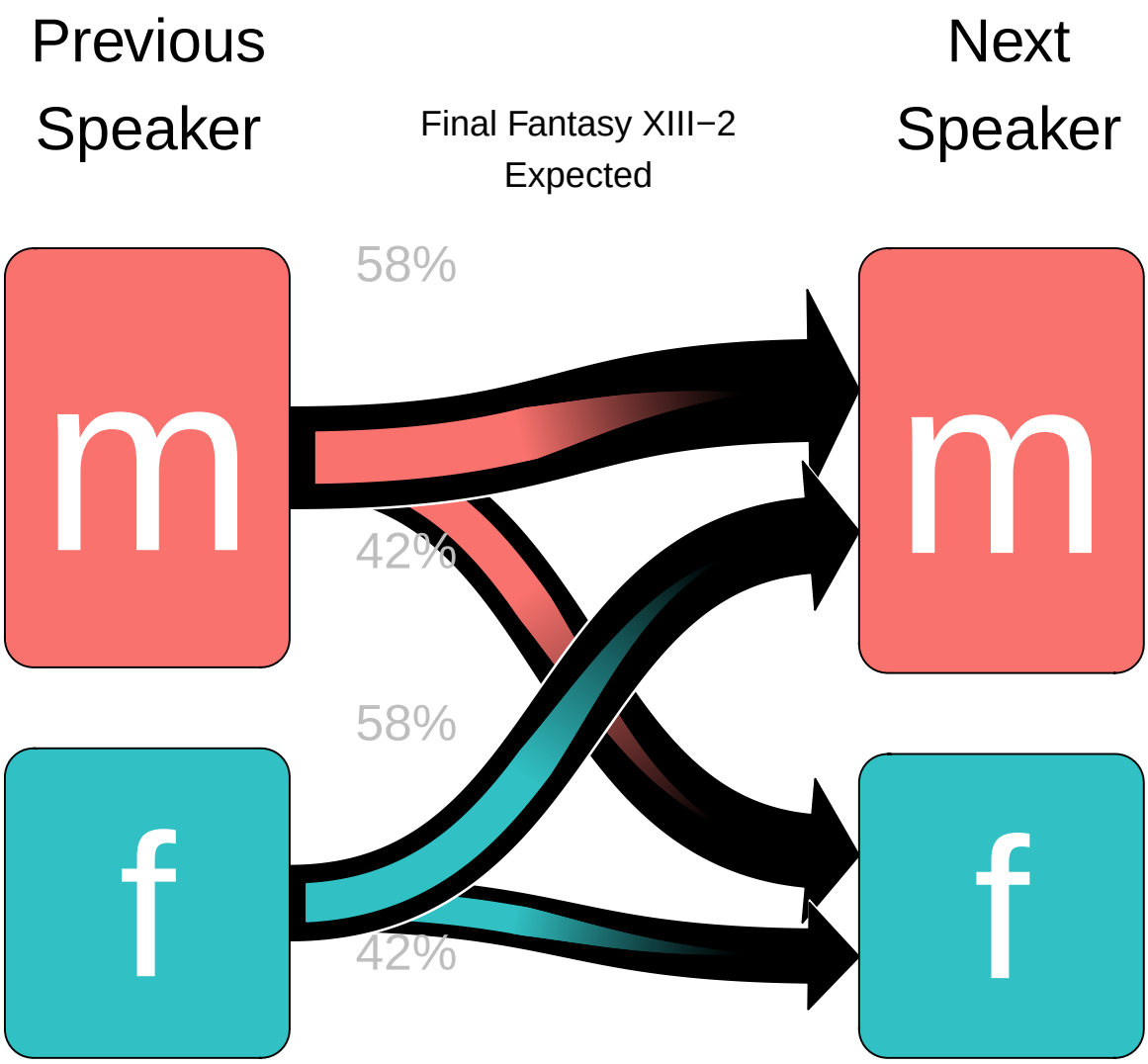

Previous Speaker
Final Fantasy XII
Empirical
Next Speaker
m
f
74%
26%
71%
29%
m
f


Previous Speaker
Final Fantasy XII Expected
Next Speaker
m
f
74%
26%
74%
26%
m
f


Previous Speaker
Final Fantasy XIII
Empirical
Next Speaker
m
f
44%
56%
68%
32%
m
f


Previous Speaker
Final Fantasy XIII
Expected
Next Speaker
m
f
54%
46%
54%
46%
m
f

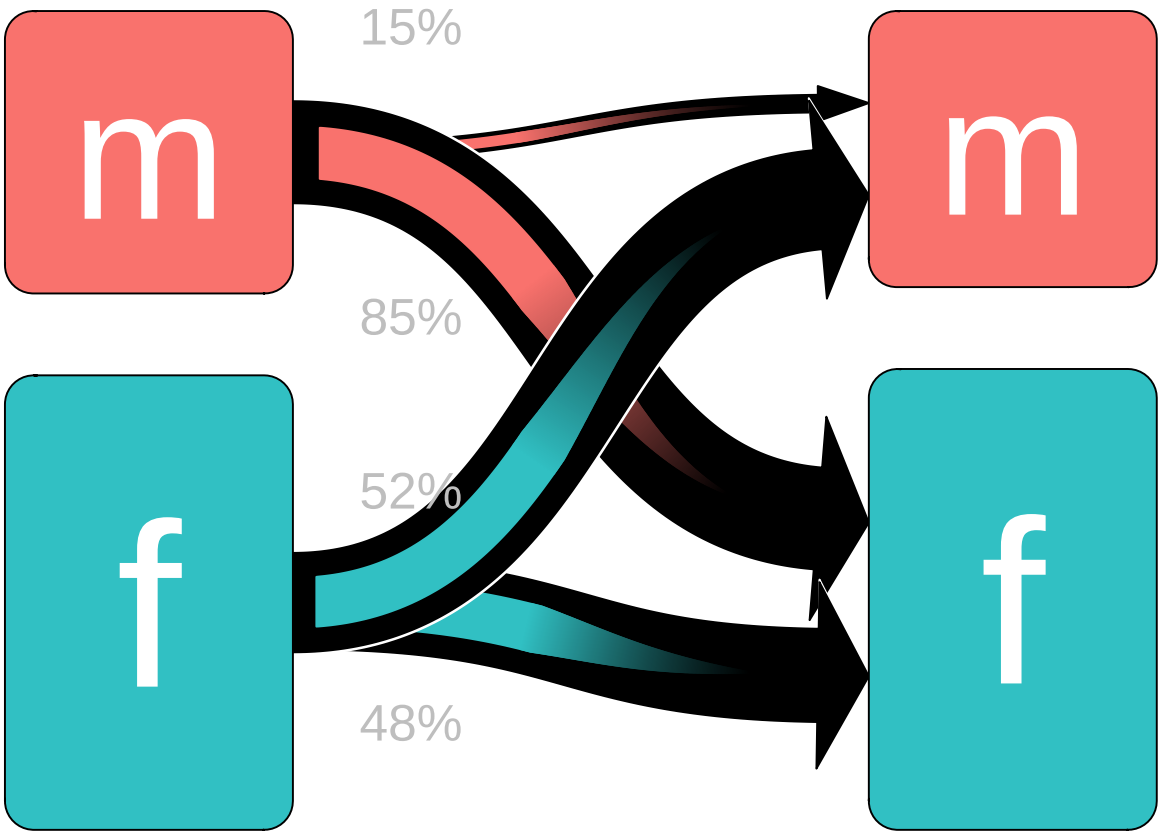

Previous Speaker
Final Fantasy XIII−2
Empirical
Next Speaker
m
f
46%
54%
78%
22%
m
f

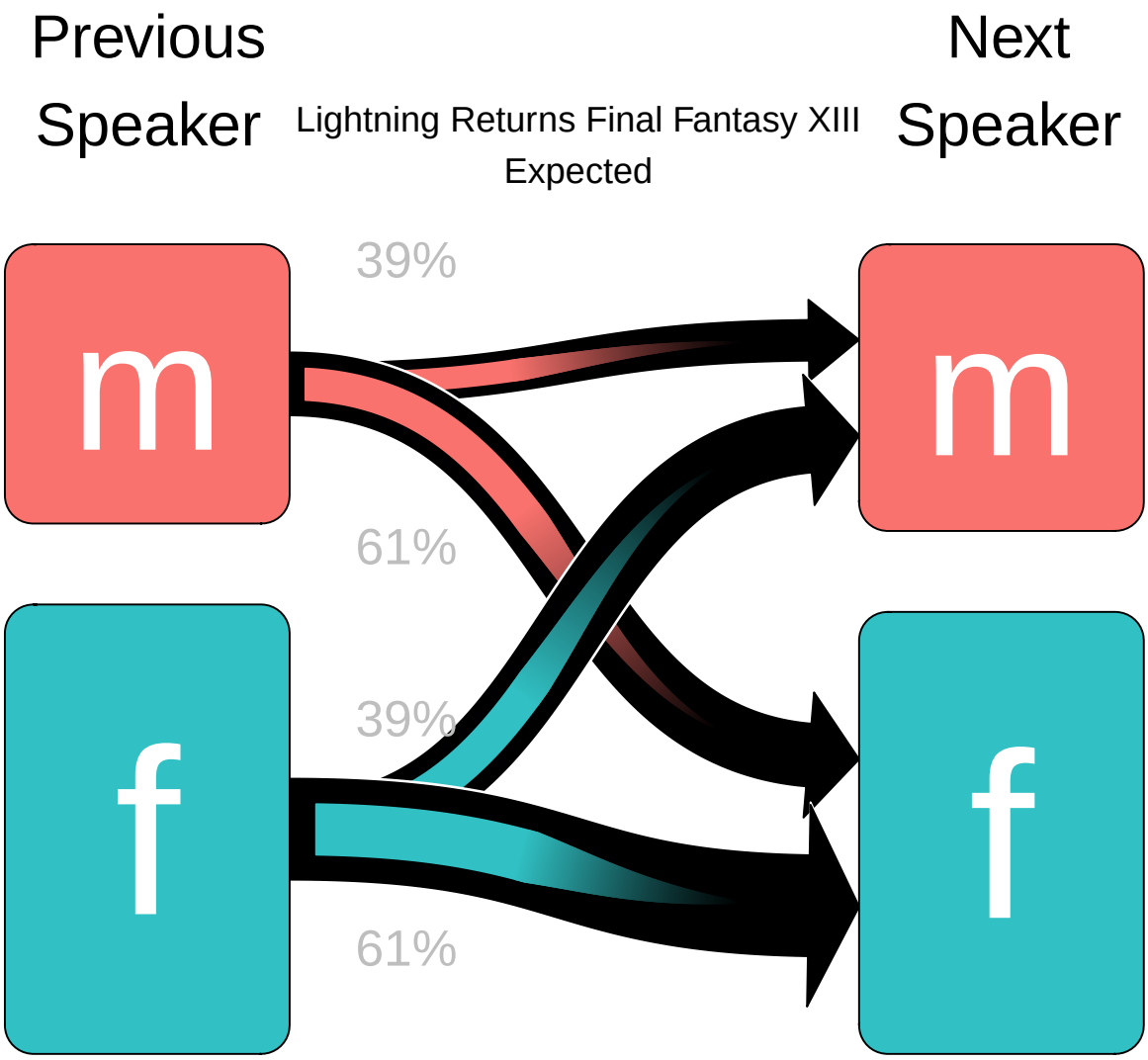

Previous Speaker
Final Fantasy XIII−2
Expected
Next Speaker
m
f
58%
42%
58%
42%
m
f

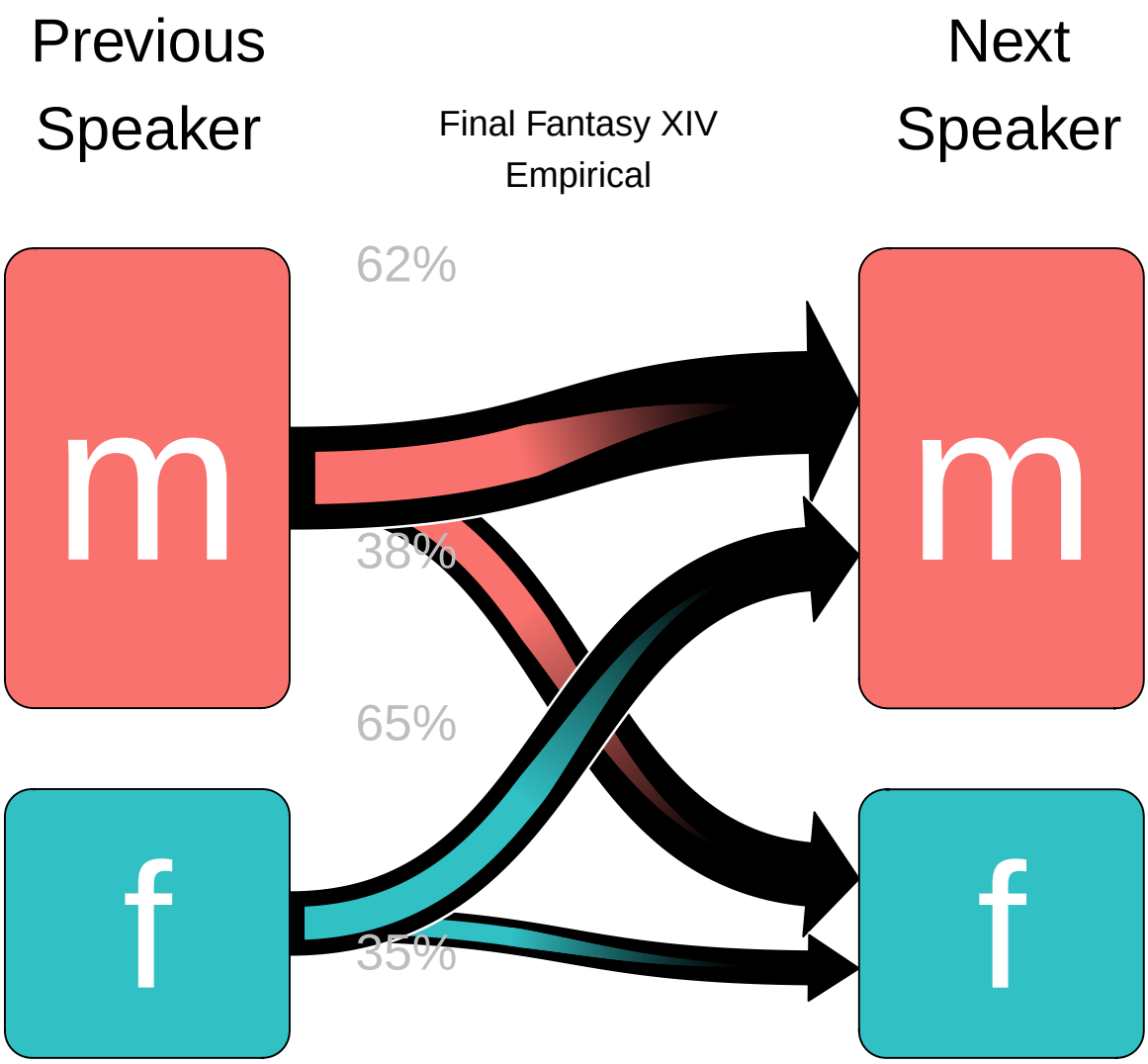
Previous Speaker

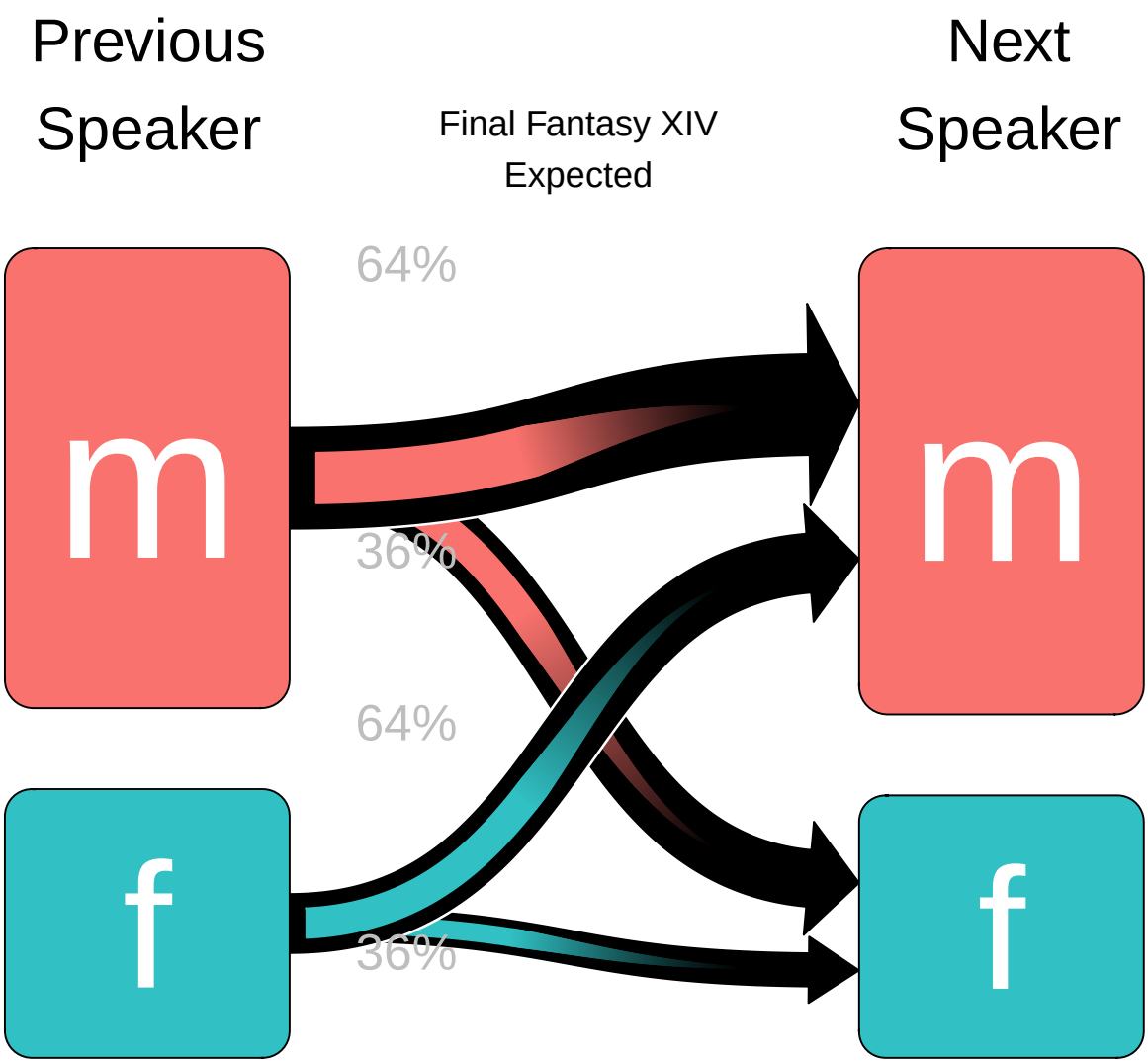
Next Speaker

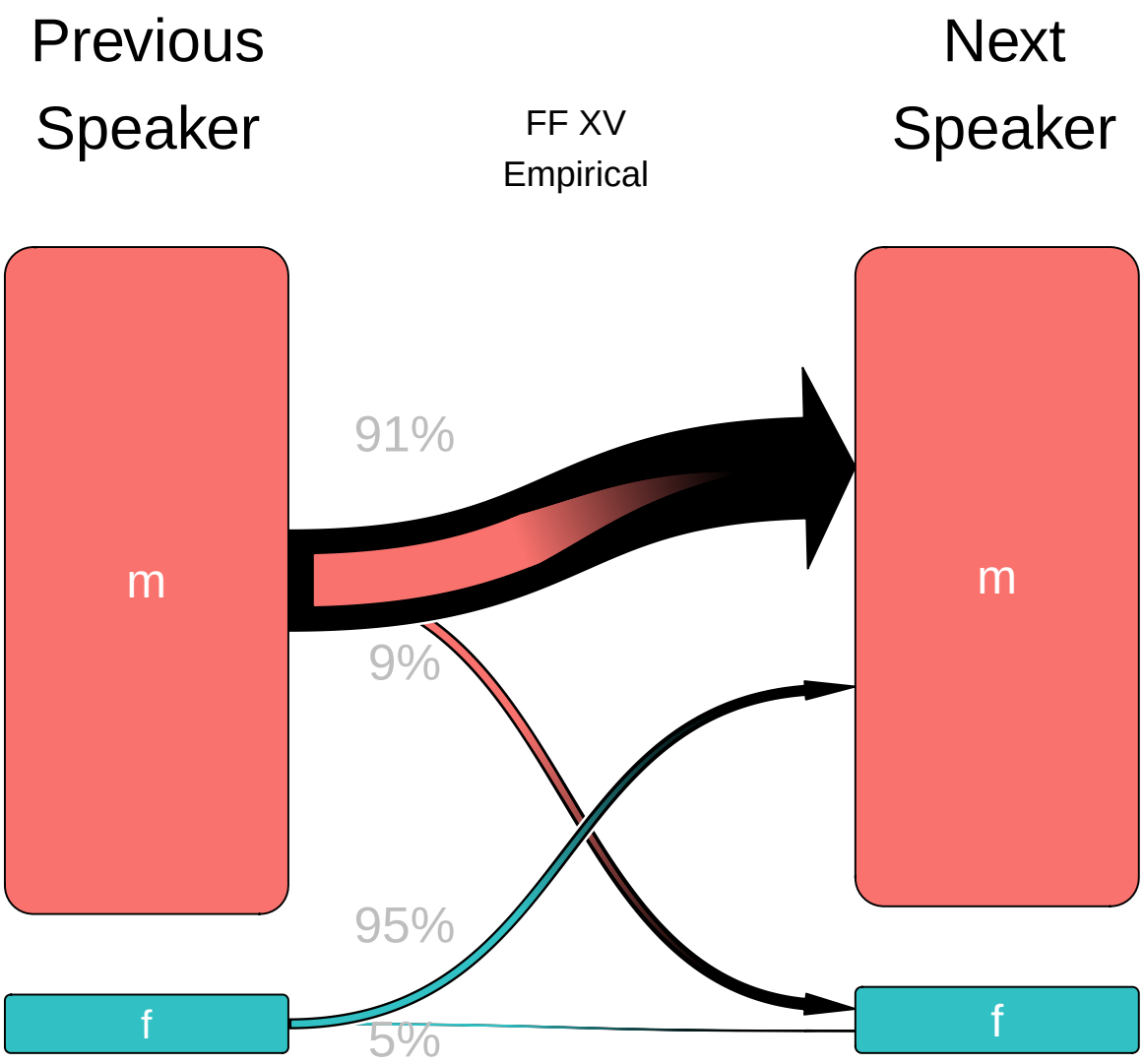
Lightning Returns Final Fantasy XIII
Empirical

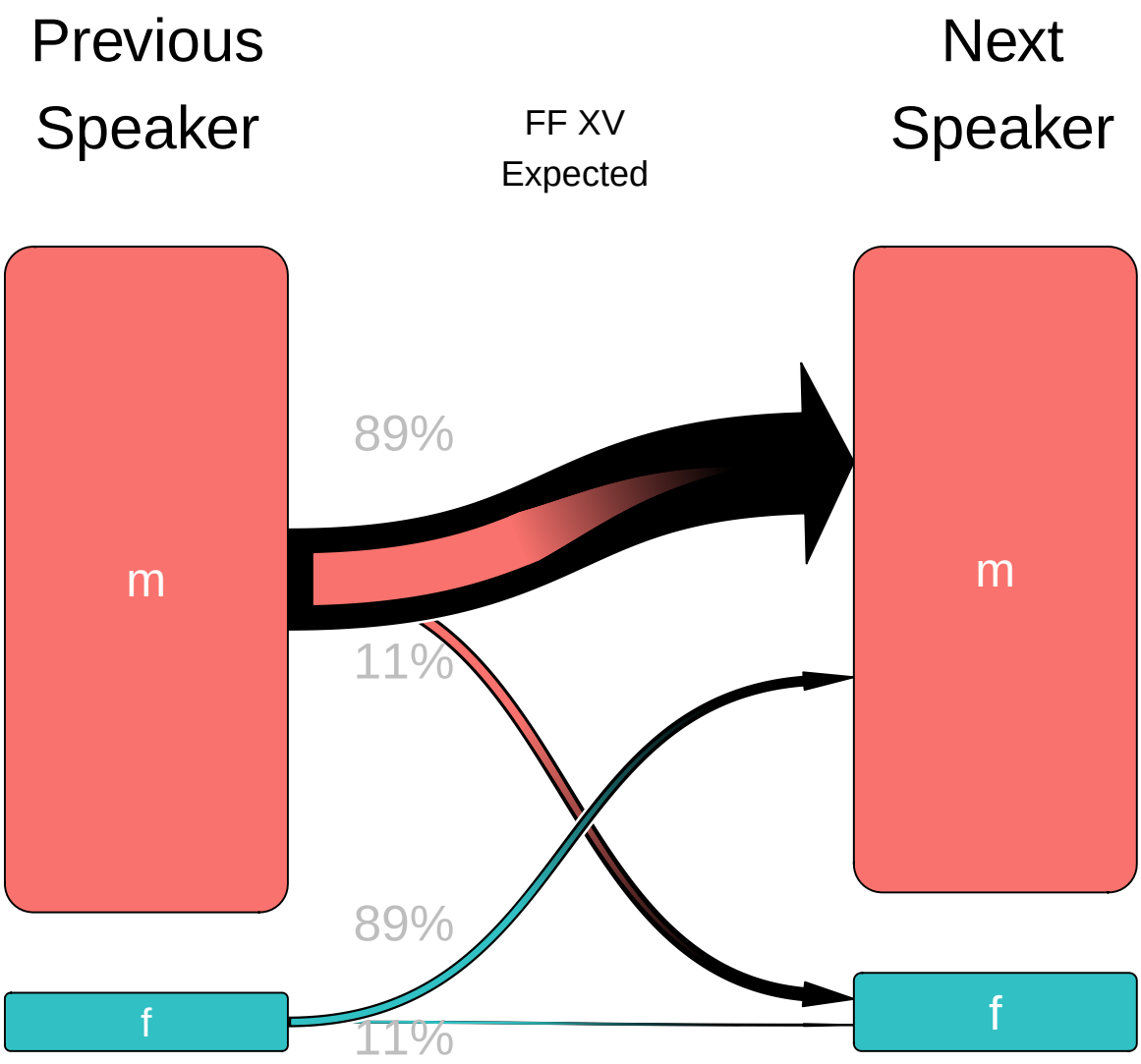


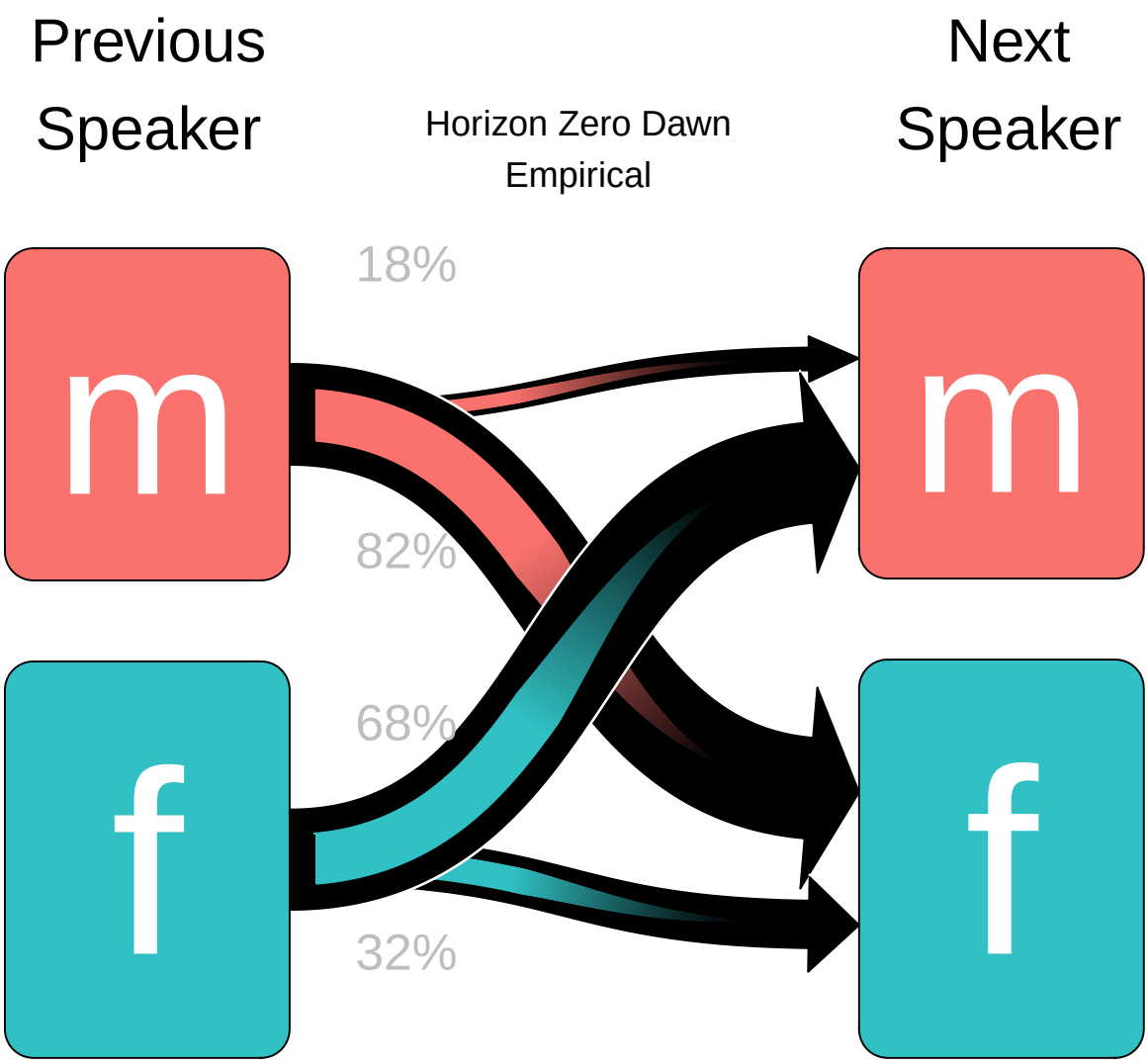

Previous Speaker
Lightning Returns Final Fantasy XIII
Expected
Next Speaker
m
f
39%
61%
39%
61%
m
f

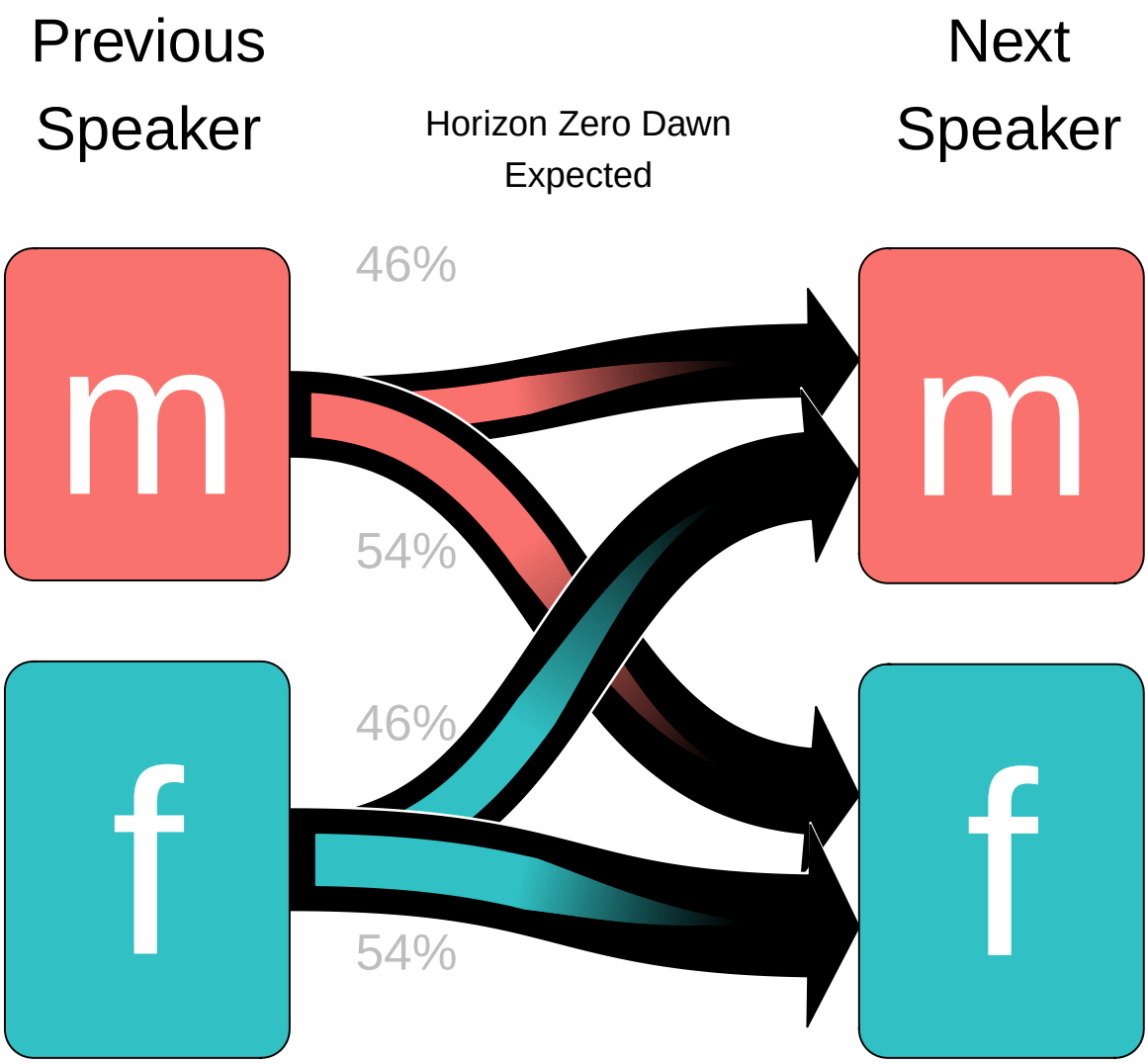

Previous Speaker
Final Fantasy XIV Empirical
Next Speaker
m
f
62%
38%
65%
35%
m
f

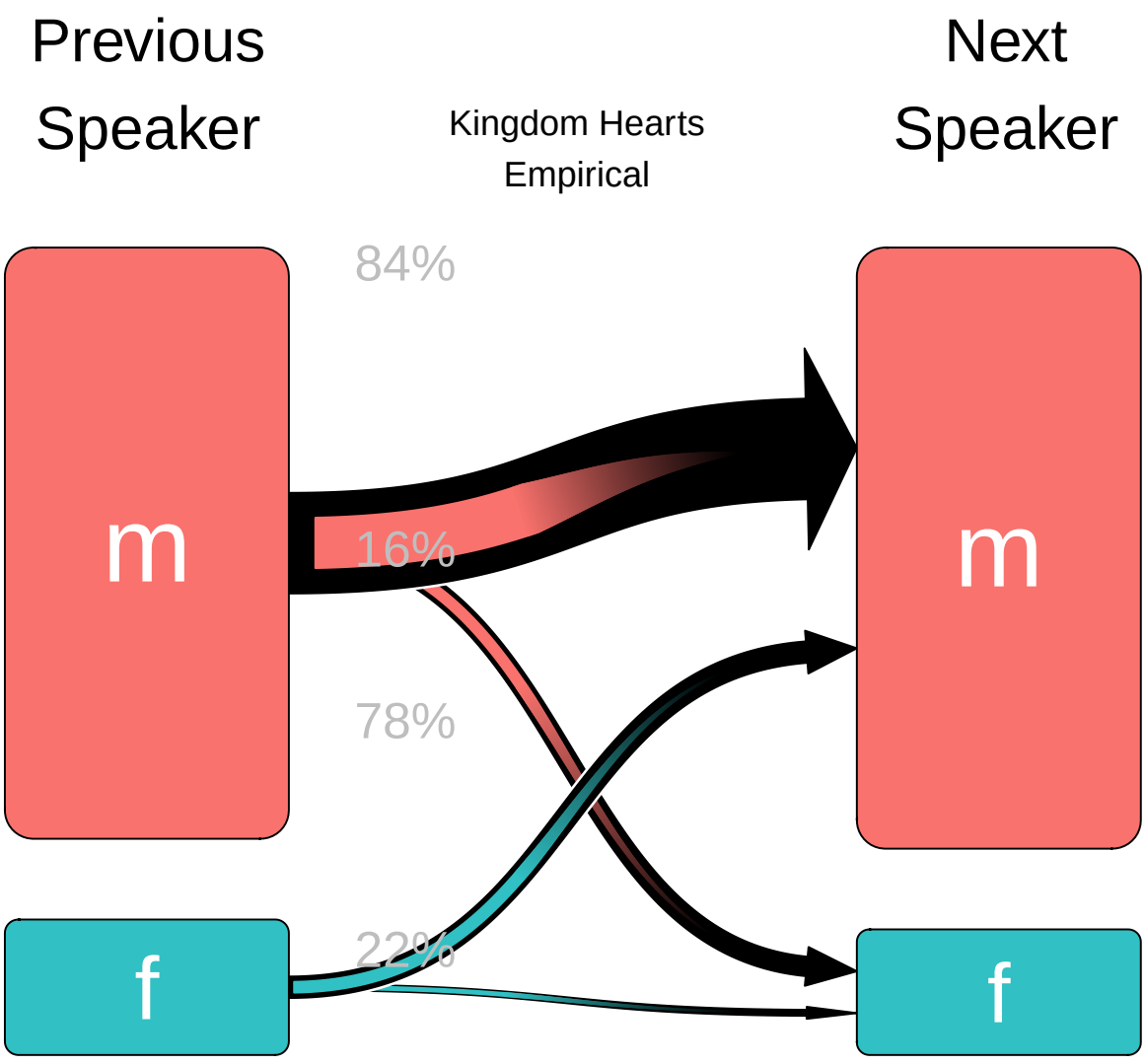

Previous Speaker
Final Fantasy XIV Expected
Next Speaker
m
f
64%
36%
64%
36%
m
f

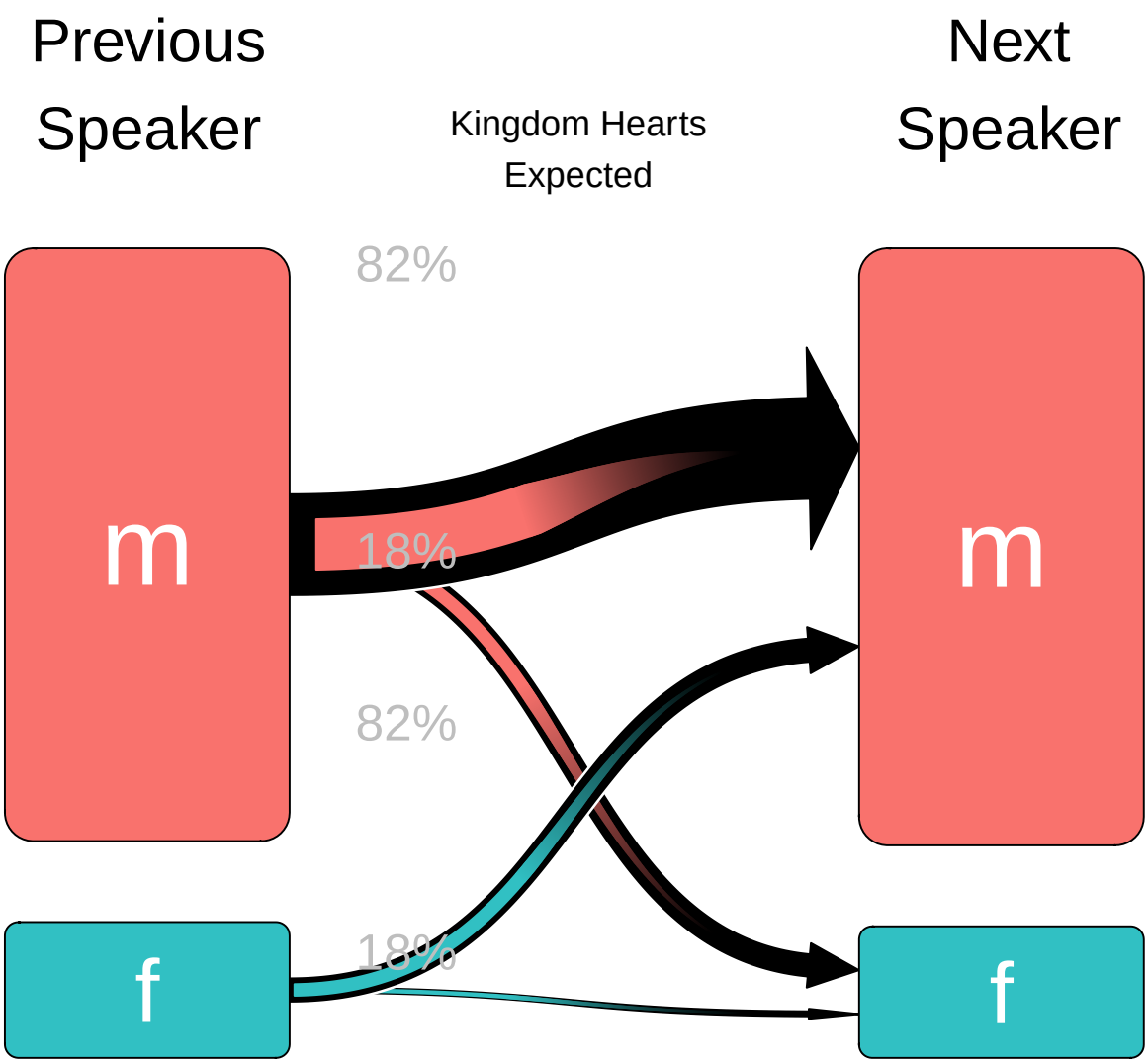

Previous Speaker
FF XV Empirical
Next Speaker
m
f
m
f
91%
9%
95%
5%

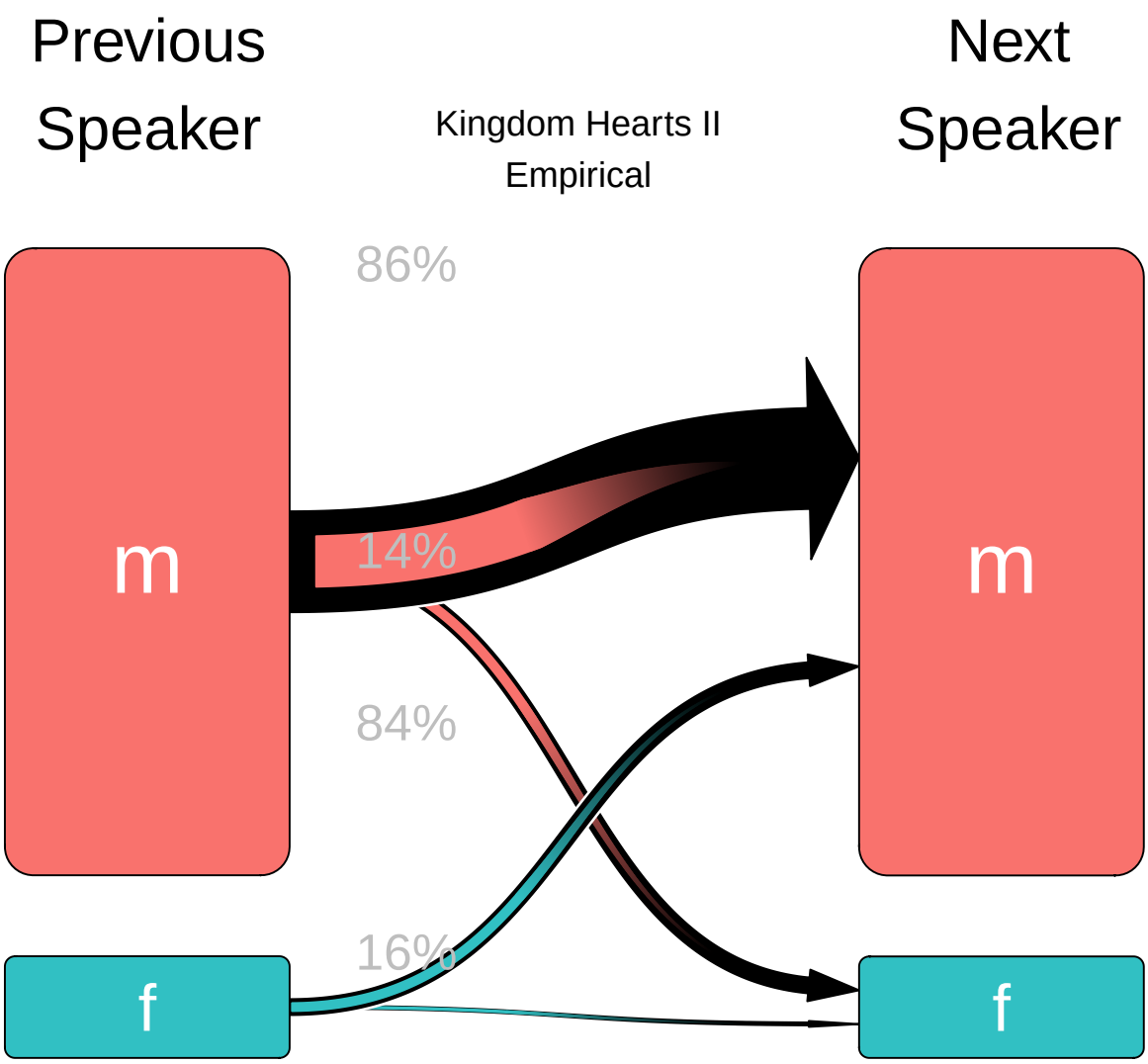

Previous Speaker
FF XV Expected
Next Speaker
m
f
89%
11%
89%
11%
m
f

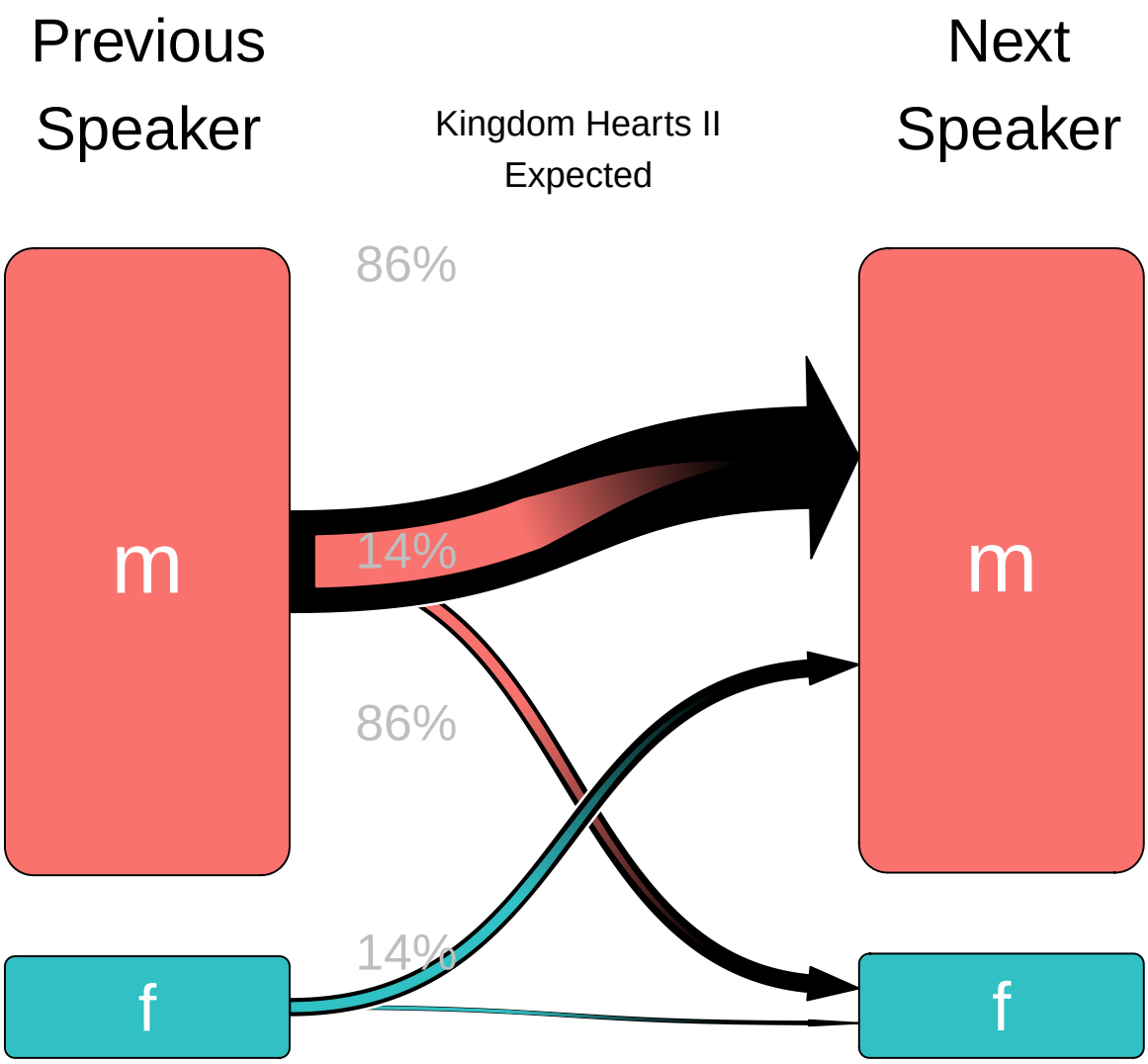

Previous Speaker
Horizon Zero Dawn Empirical
Next Speaker
m
f
18%
82%
68%
32%
m
f

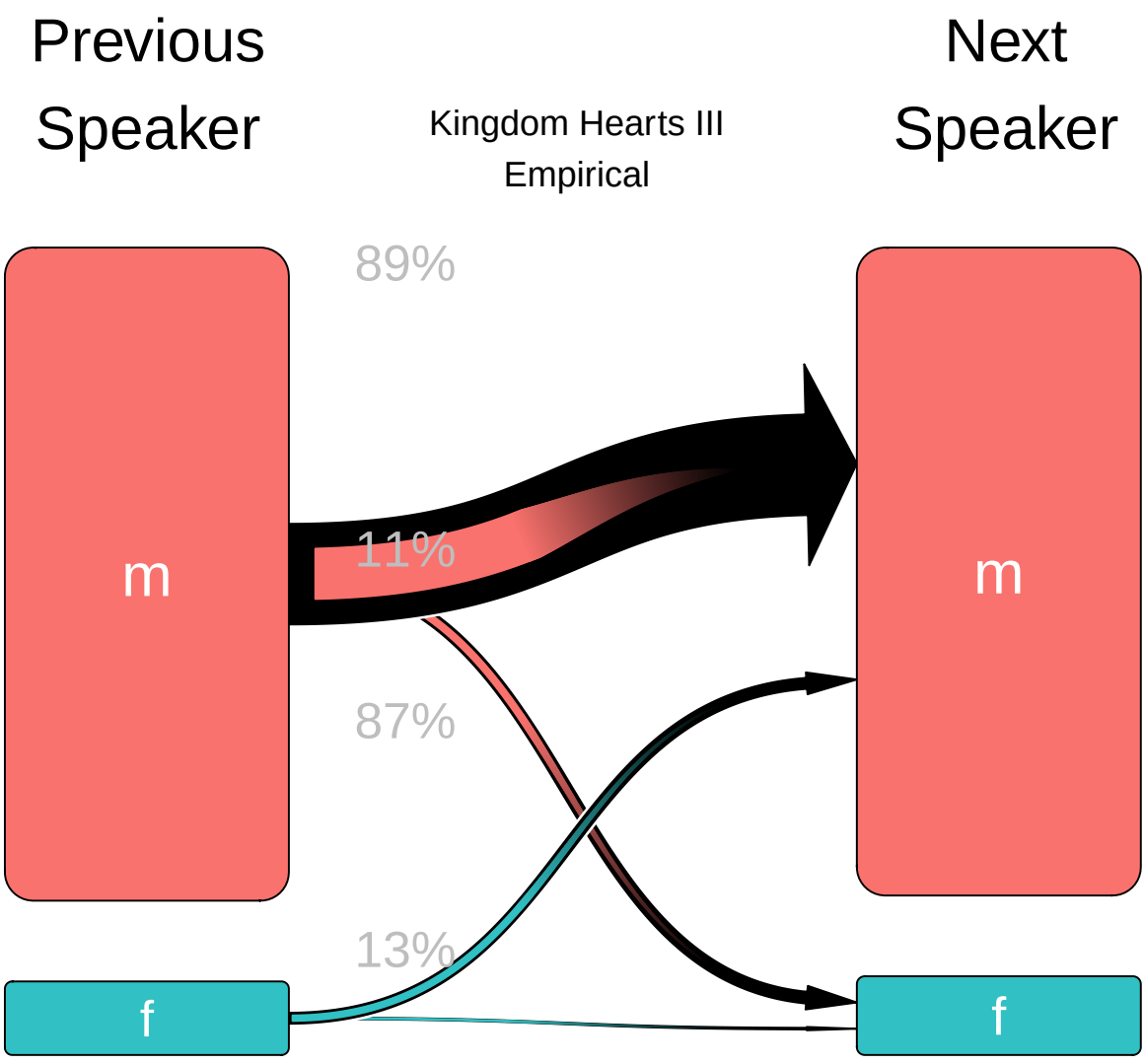

Previous Speaker
Horizon Zero Dawn Expected
Next Speaker
m
f
46%
54%
46%
54%
m
f

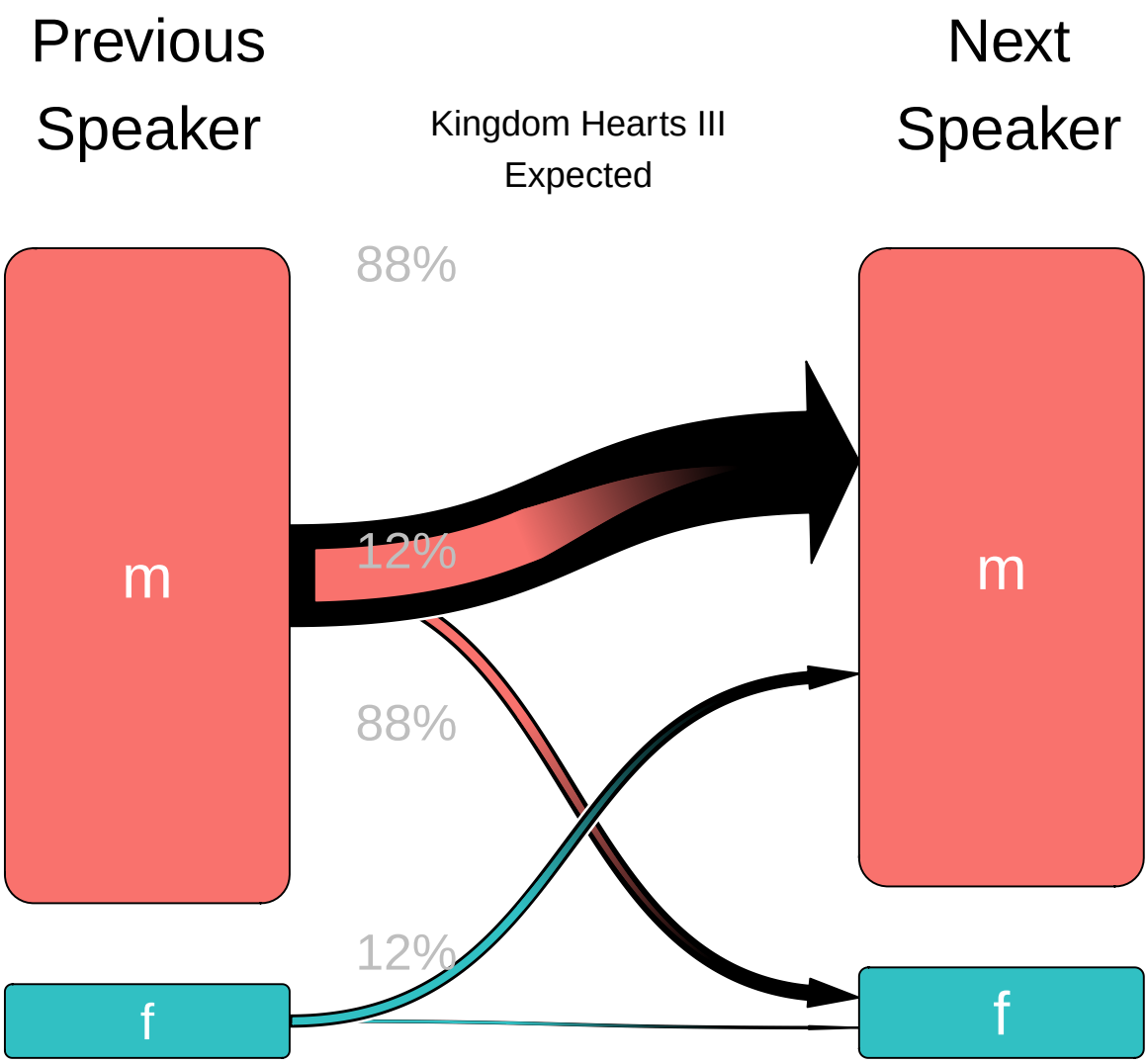

Previous Speaker
Kingdom Hearts Empirical
Next Speaker
m
f
84%
16%
78%
22%
m
f


Previous Speaker
Kingdom Hearts Expected
Next Speaker
m
f
82%
18%
82%
18%
m
f


Previous Speaker
Kingdom Hearts II Empirical
Next Speaker
m
f
86%
14%
84%
16%
m
f


Previous Speaker
Kingdom Hearts II Expected
Next Speaker
m
f
86%
14%
86%
14%
m
f

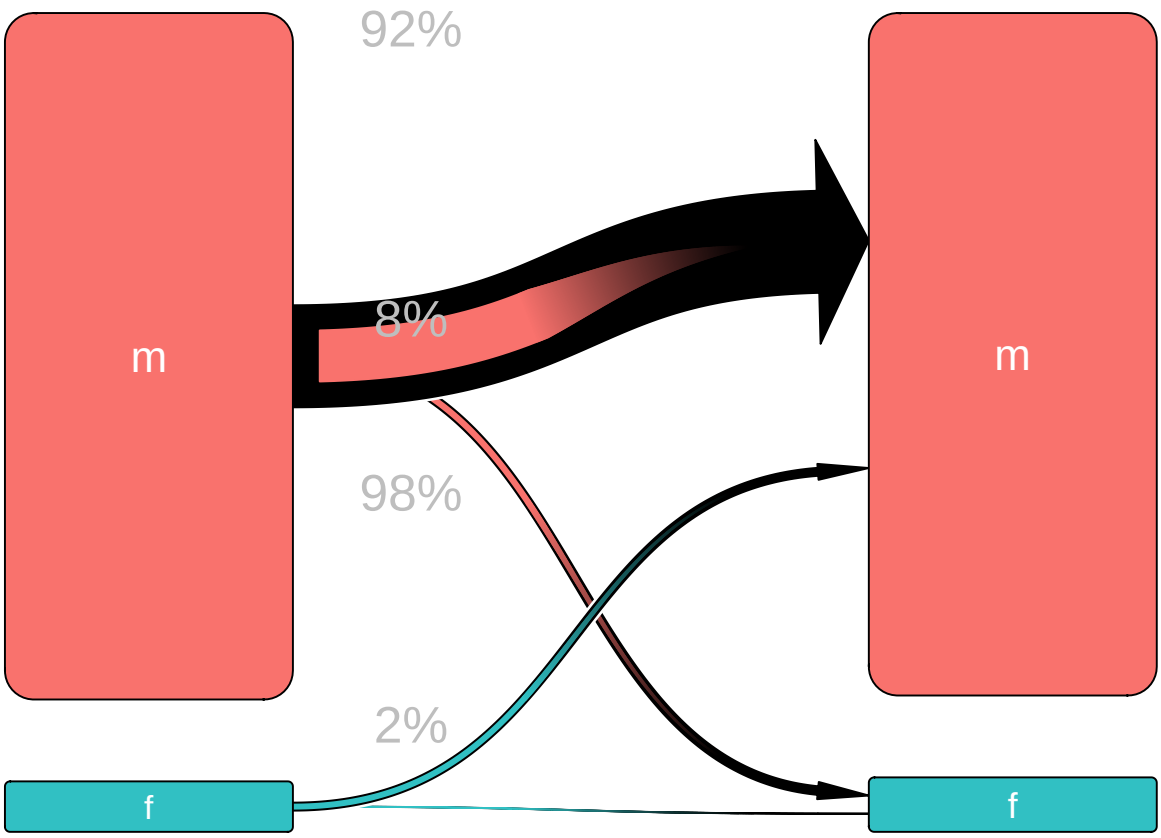

Previous Speaker
Kingdom Hearts III
Empirical
Next Speaker
m
f
89%
11%
87%
13%
m
f


Previous Speaker
Kingdom Hearts III Expected
Next Speaker
m
f
88%
12%
88%
12%
m
f

Previous Speaker

Next Speaker

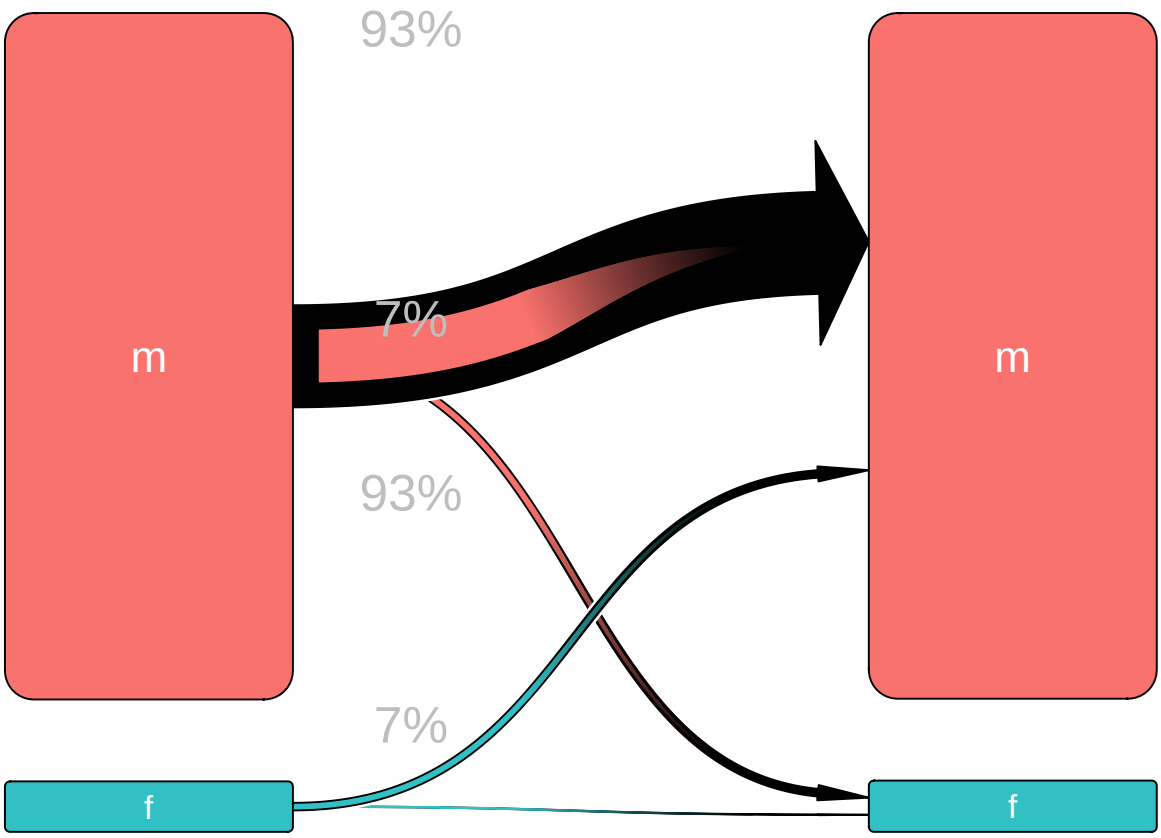
Kingdom Hearts 3D Dream Drop Distance
Empirical

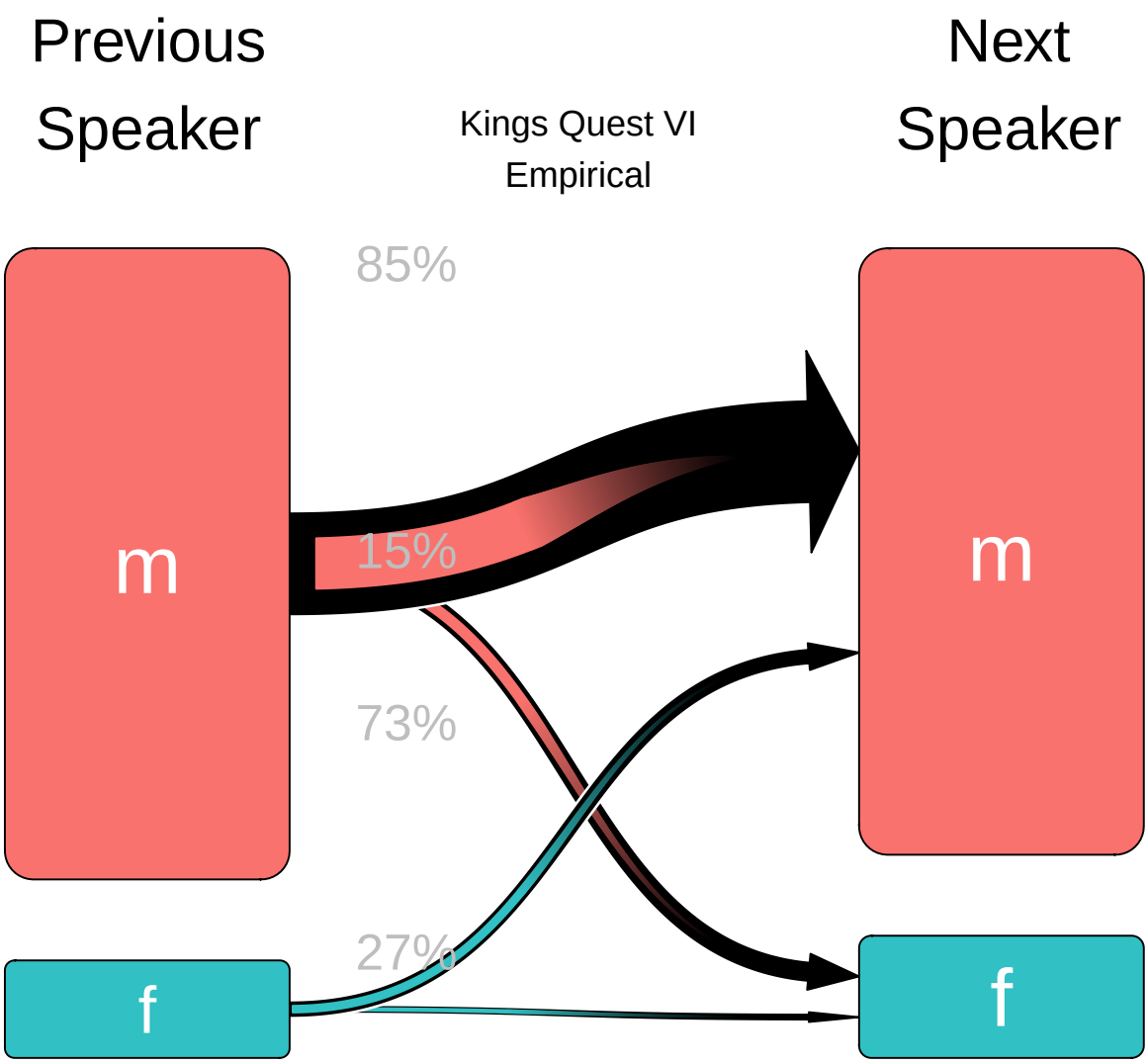


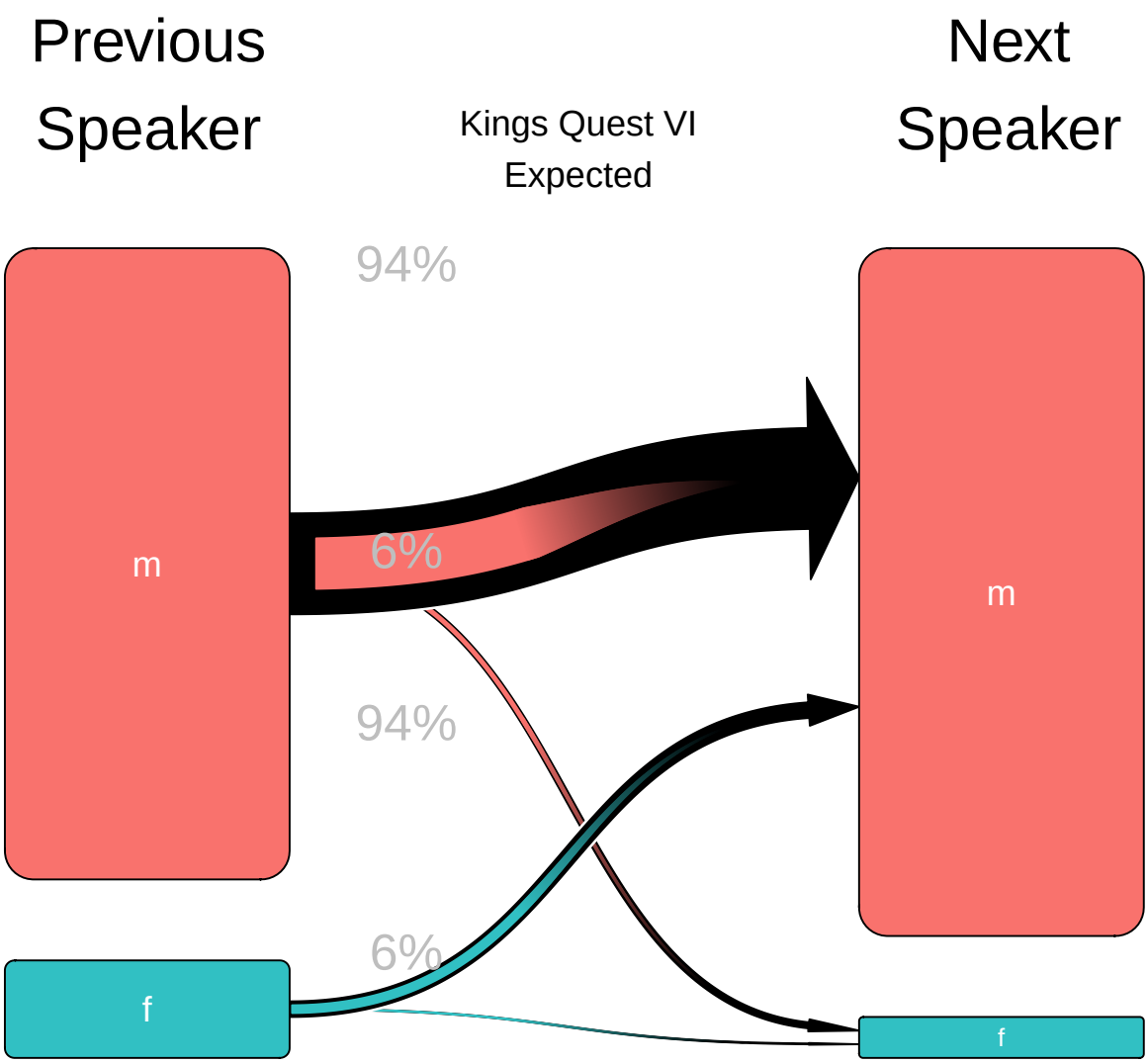
Previous Speaker

Next Speaker

Kingdom Hearts 3D Dream Drop Distance
Expected



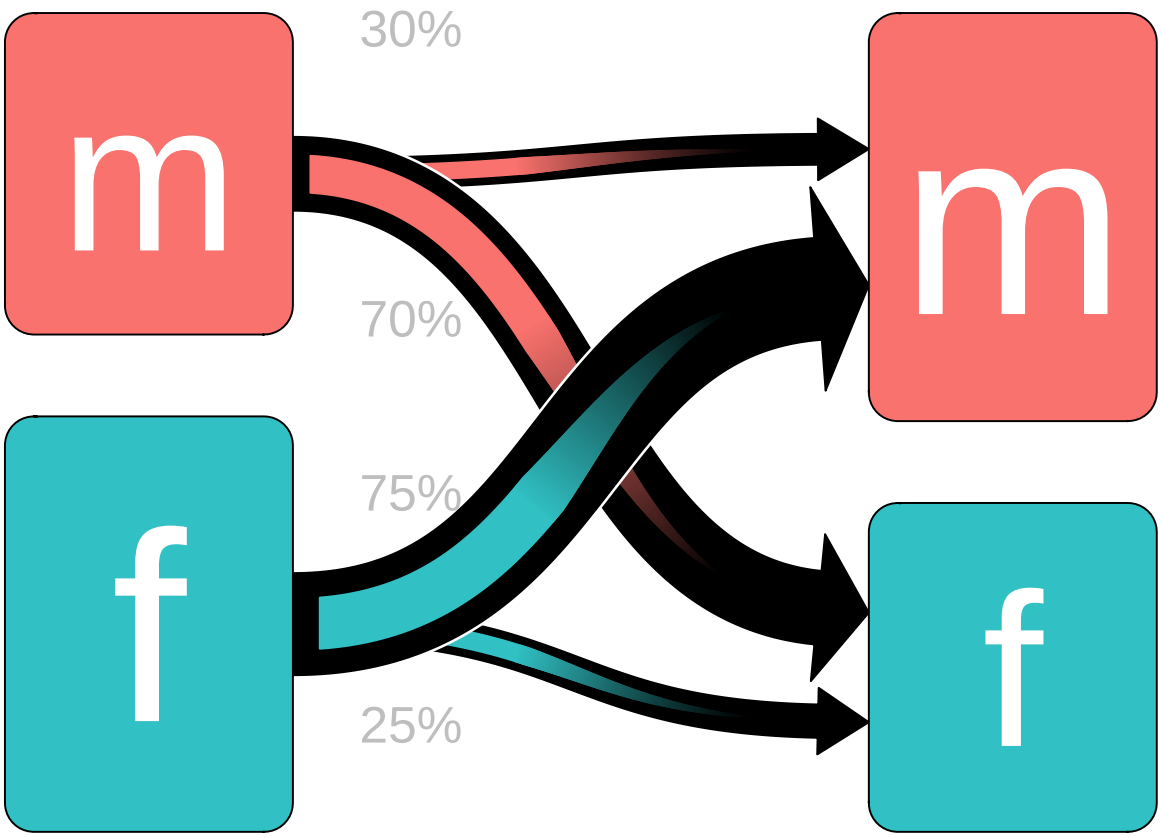

Previous Speaker
Kings Quest VI Empirical
Next Speaker
m
f
85%
15%
73%
27%
m
f


Previous Speaker
Kings Quest VI
Expected
Next Speaker
94%
6%
94%
6%
m
f
m
f

Previous Speaker

Next Speaker

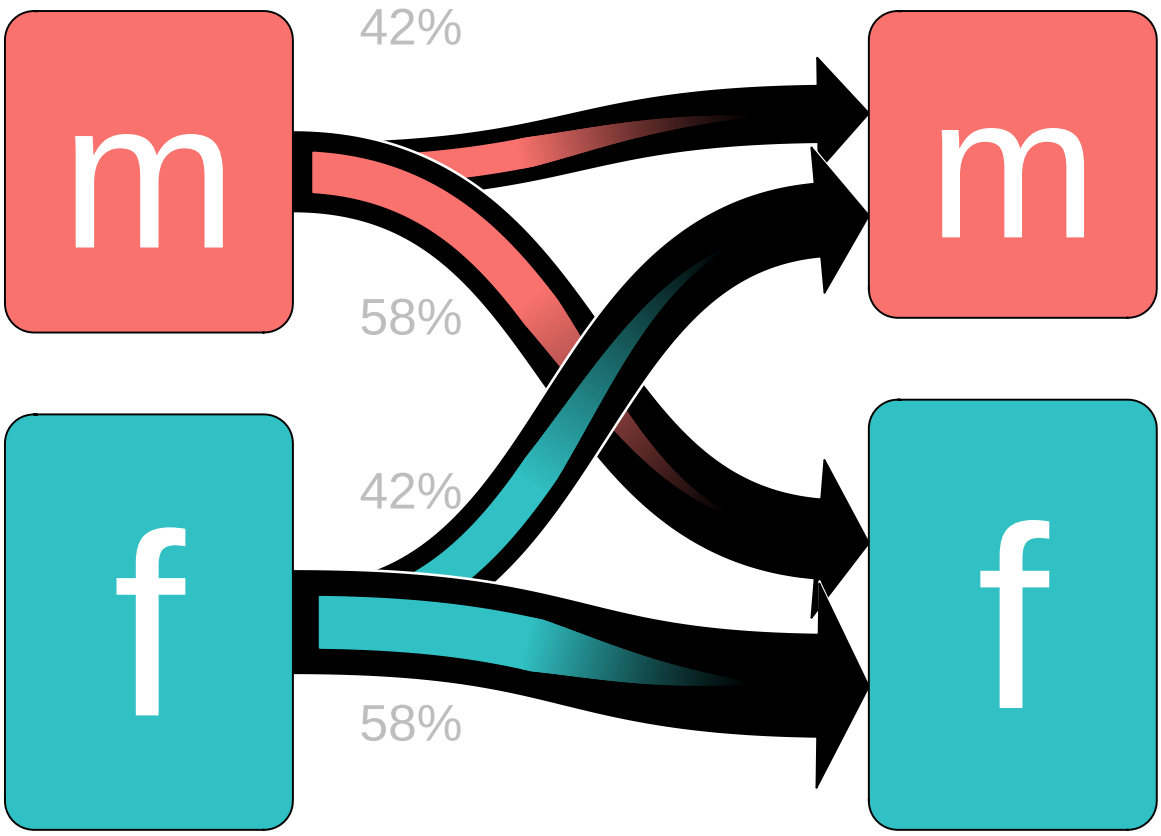
Kings Quest VII The Princeless Bride
Empirical

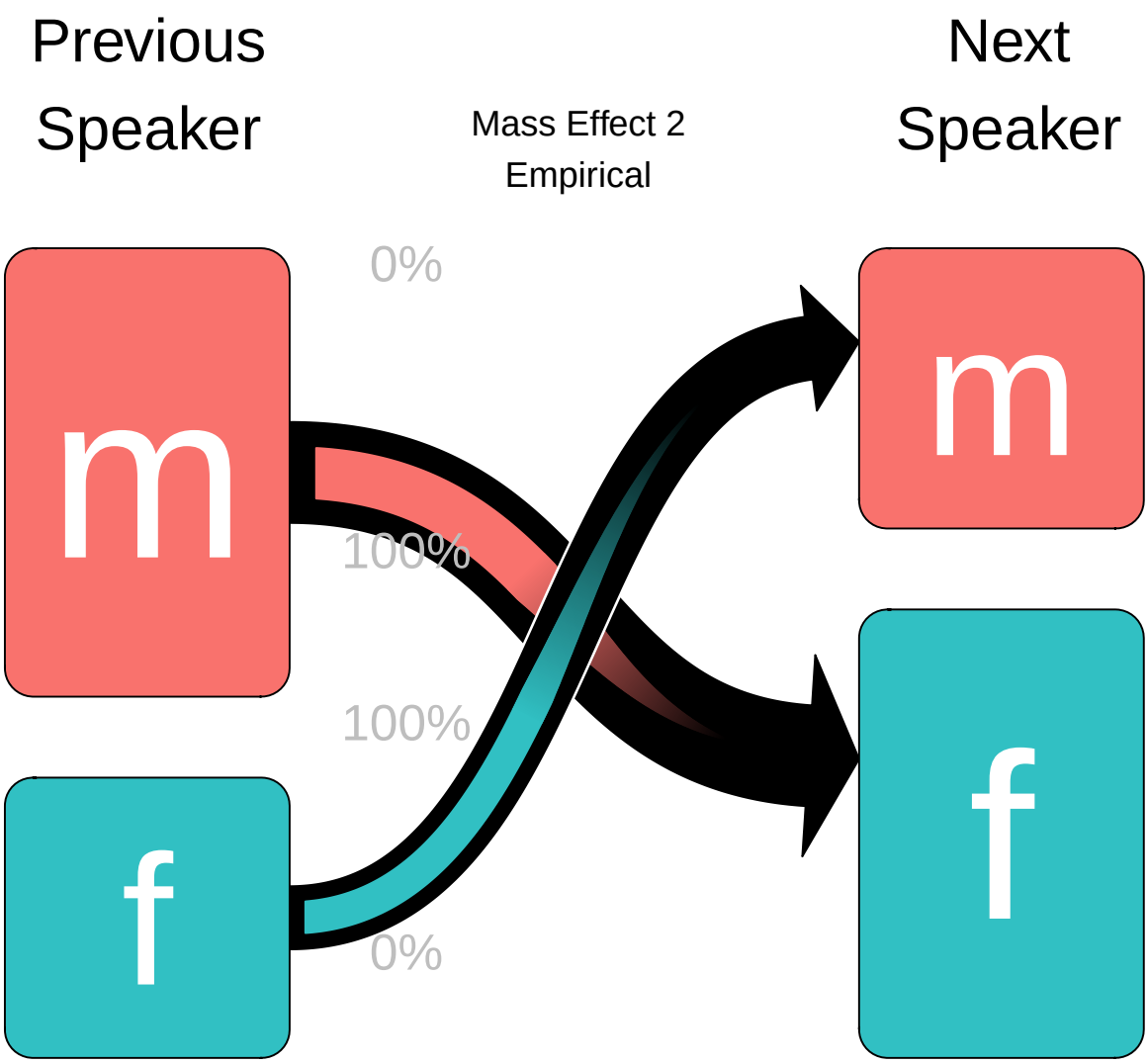


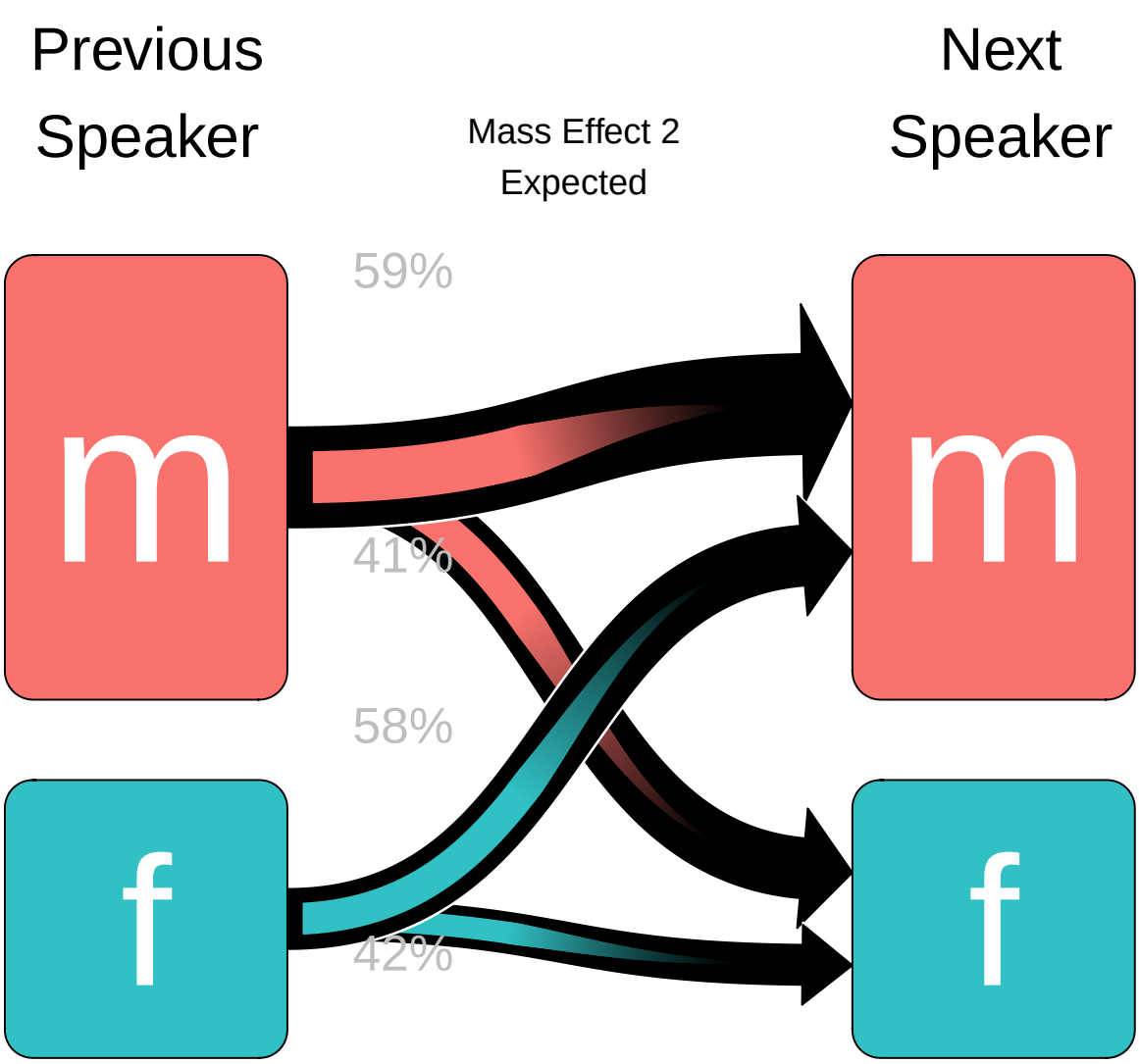
Previous Speaker

Next Speaker

Kings Quest VII The Princeless Bride Expected



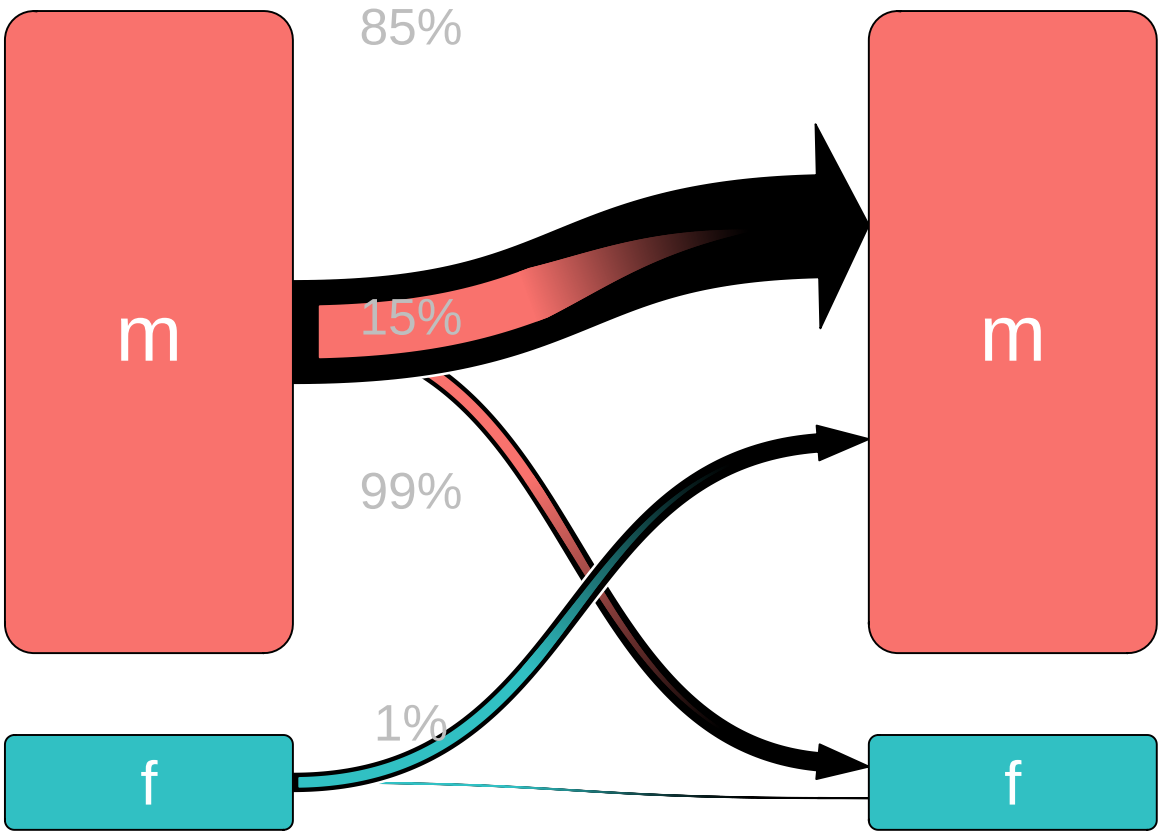

Previous Speaker
Mass Effect 2 Empirical
Next Speaker
m
f
0%
100%
100%
0%
m
f

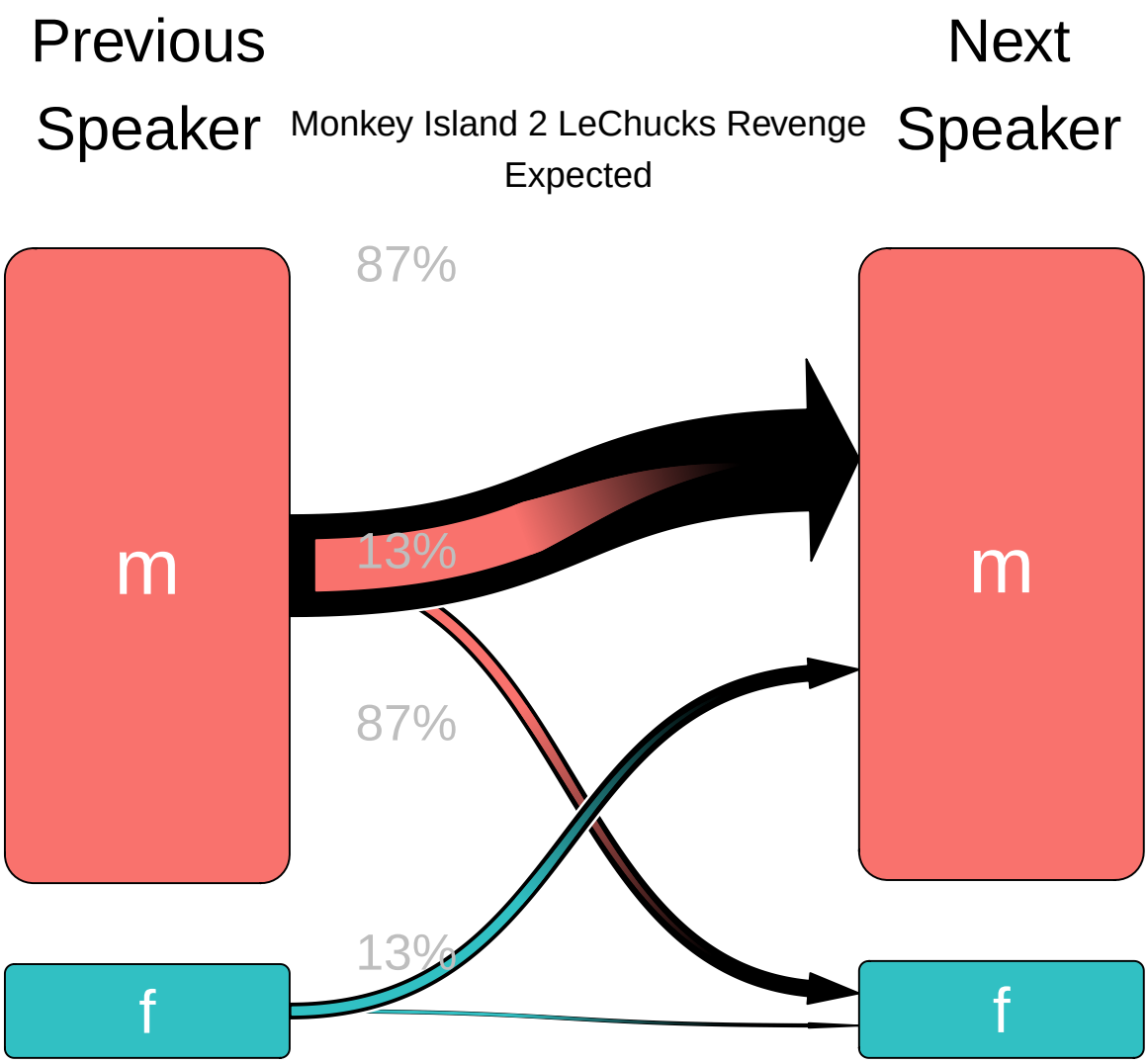

Previous Speaker
Mass Effect 2 Expected
Next Speaker
m
f
59%
41%
58%
42%
m
f

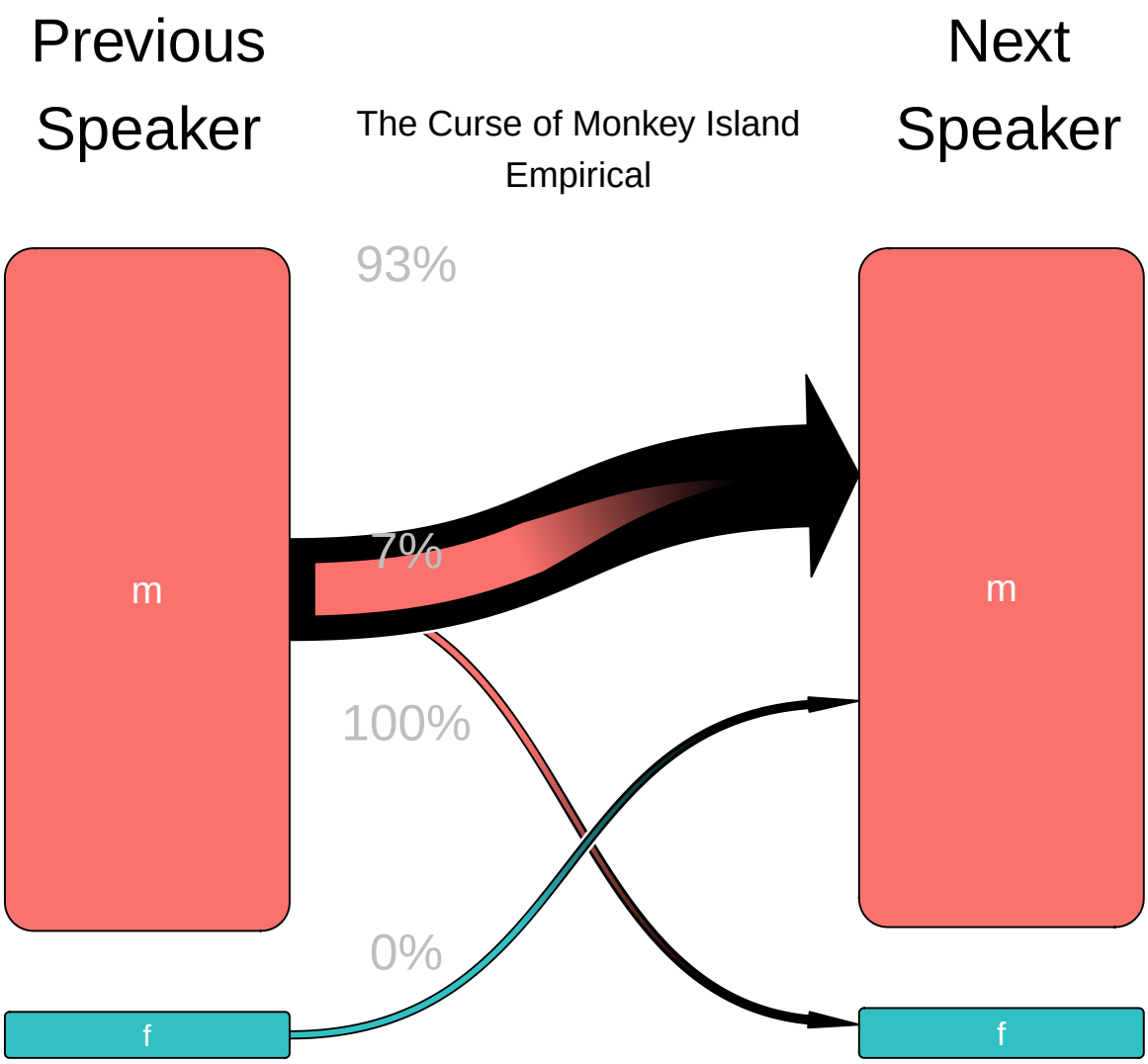
Previous Speaker

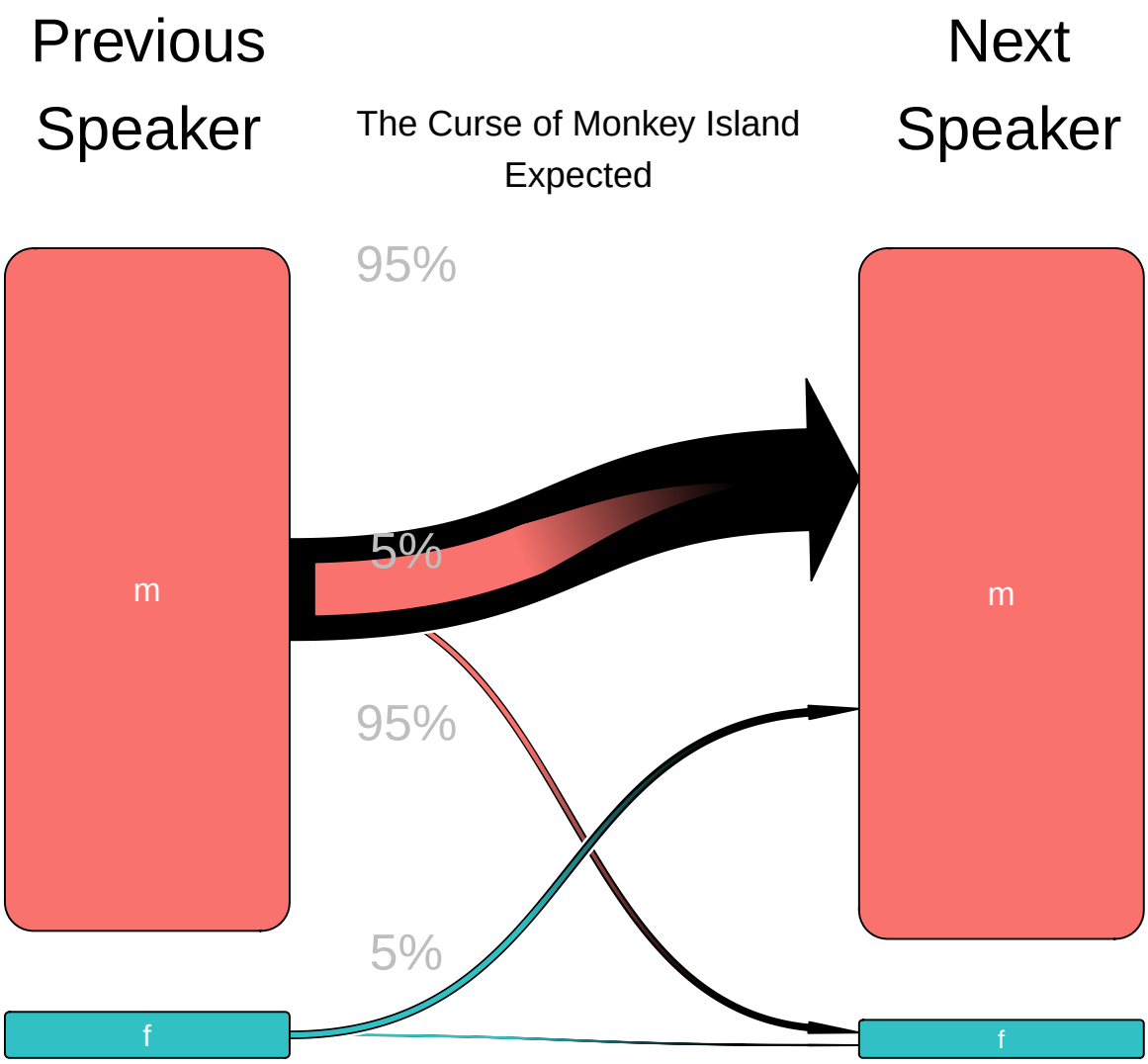
Next Speaker

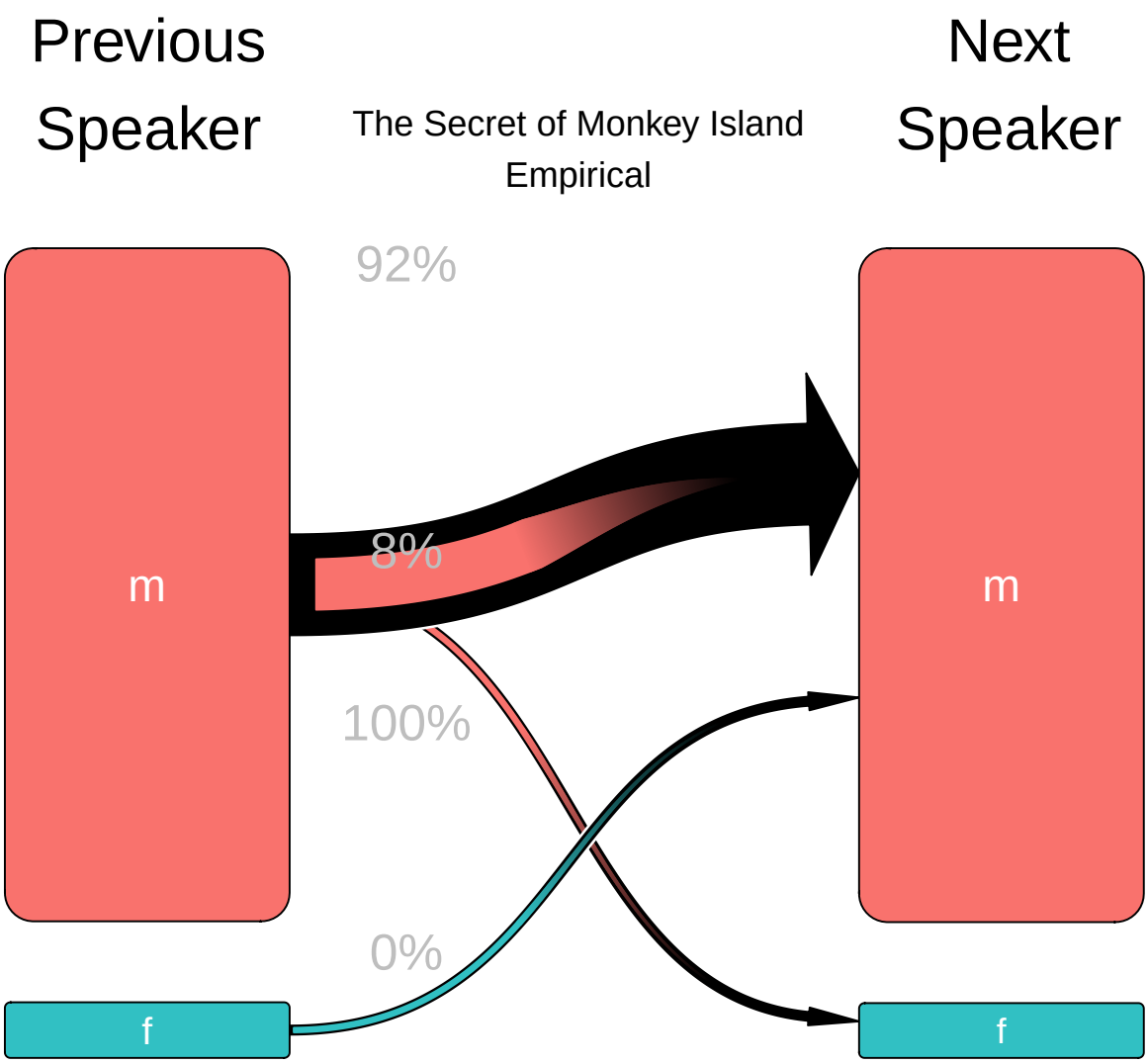
Monkey Island 2 LeChucks Revenge
Empirical

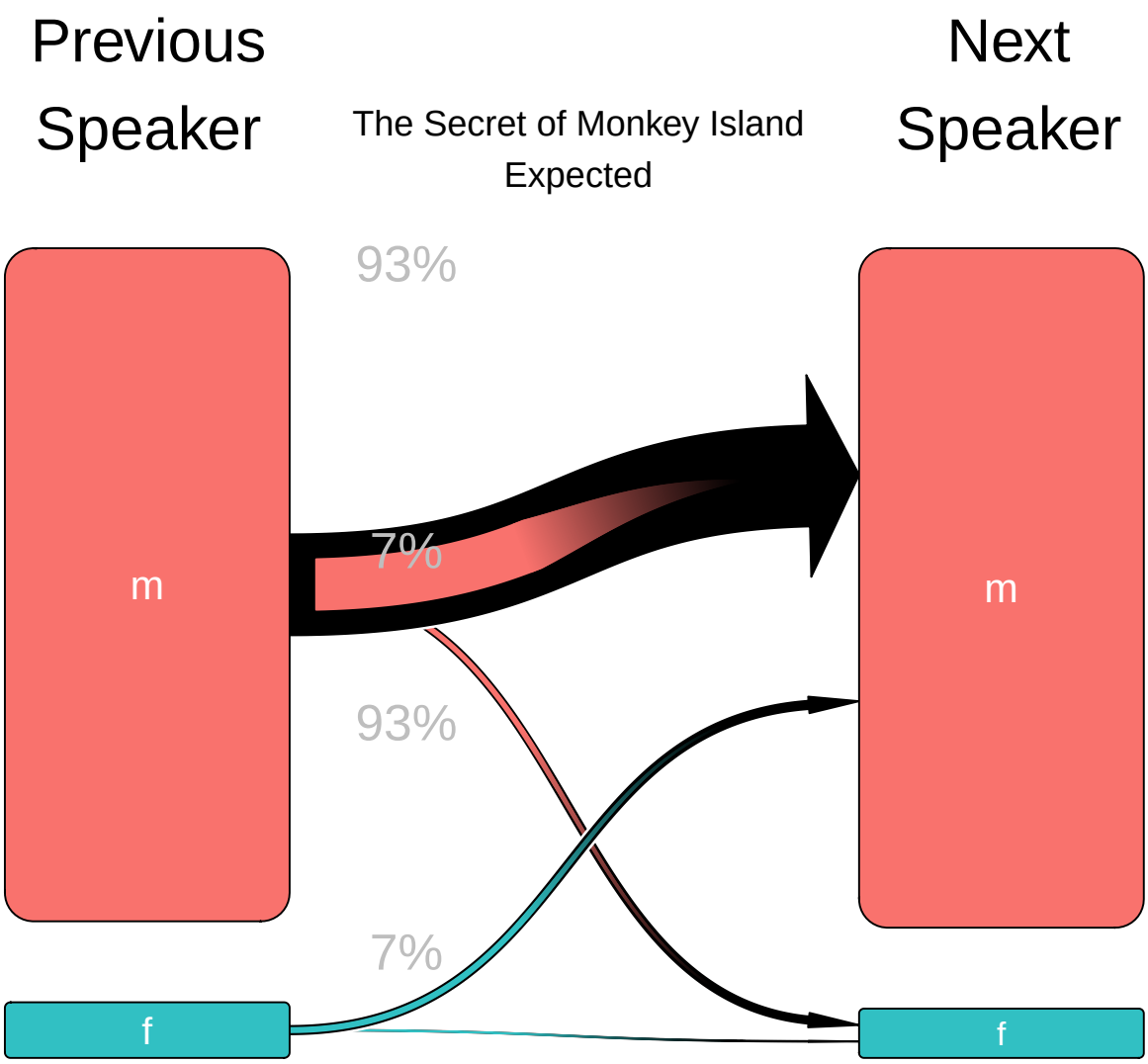


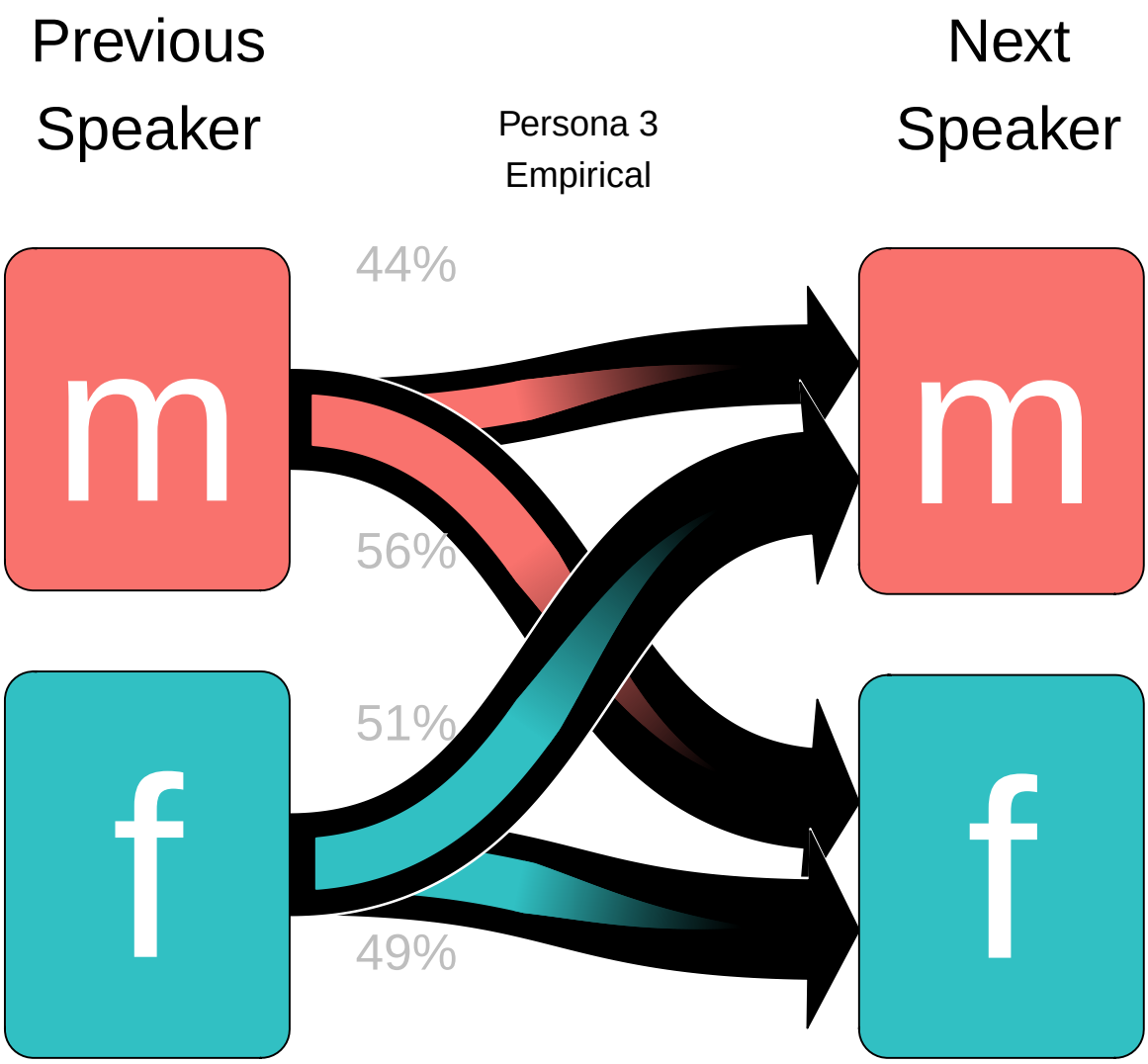

Previous Speaker
Monkey Island 2 LeChucks Revenge
Expected
Next Speaker
m
f
87%
13%
87%
13%
m
f

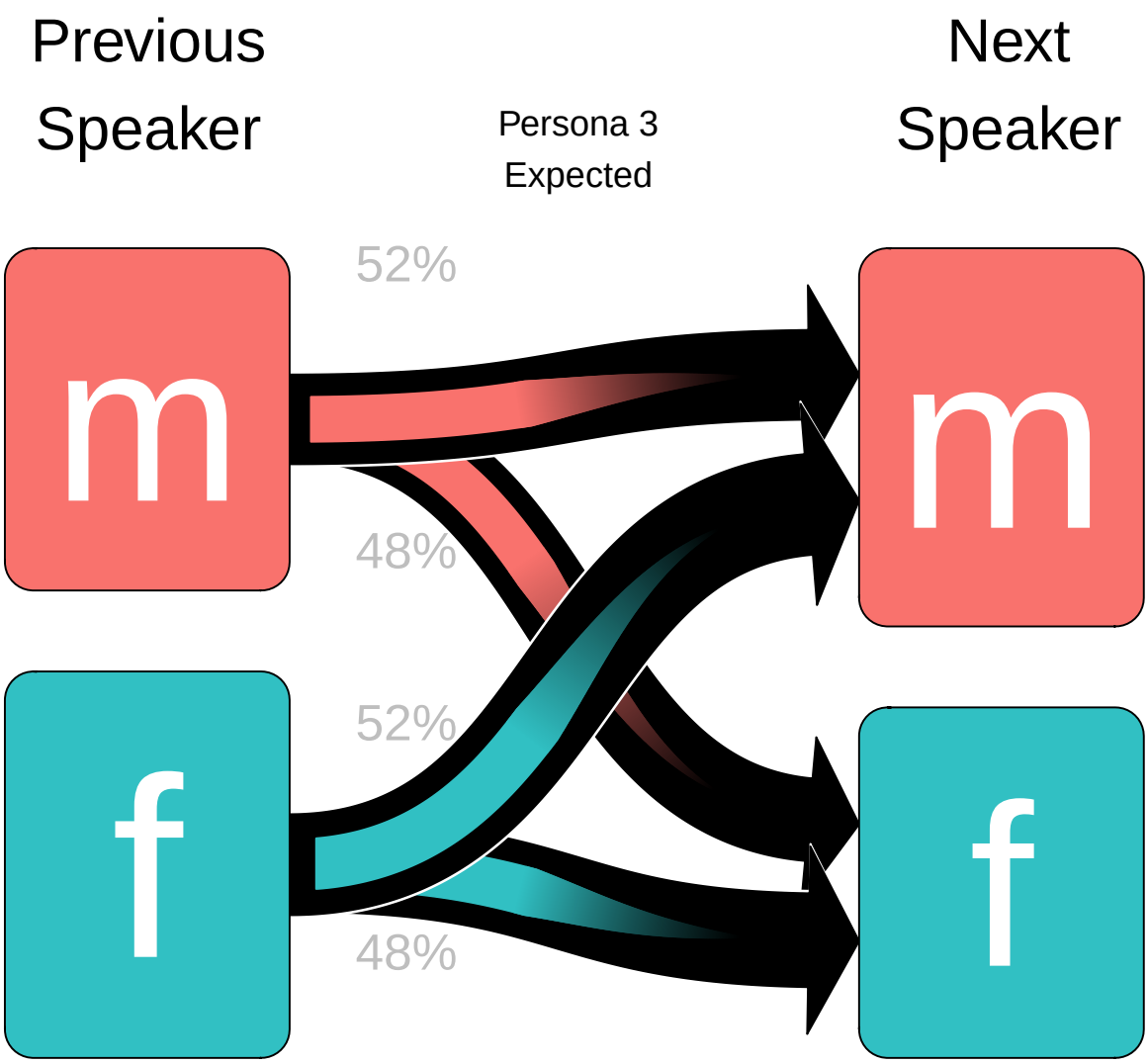

Previous Speaker
The Curse of Monkey Island
Empirical
Next Speaker
93%
7%
100%
0%
m
f
m
f

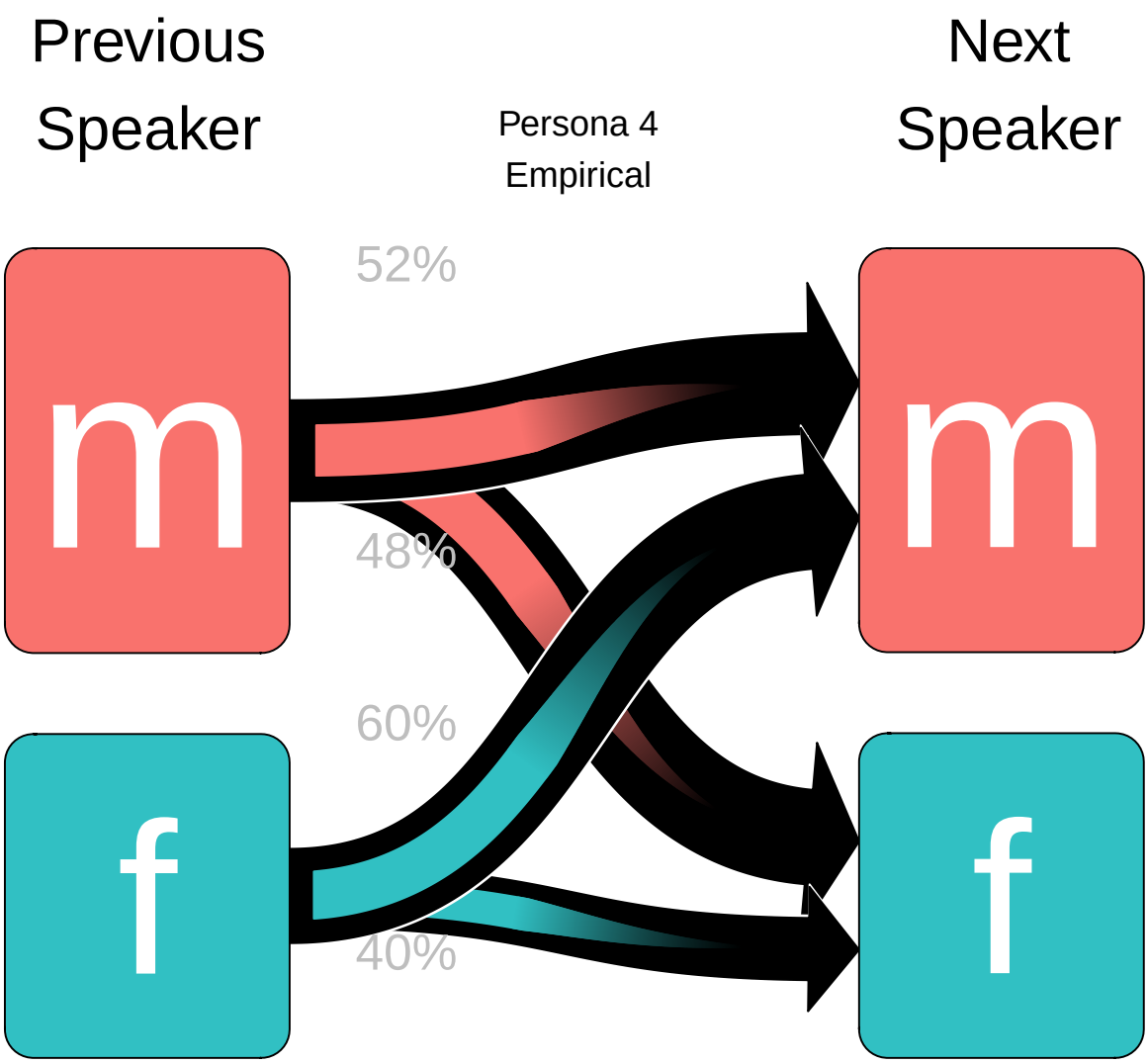

Previous Speaker
The Curse of Monkey Island
Expected
Next Speaker
95%
5%
95%
5%
m
f
m
f

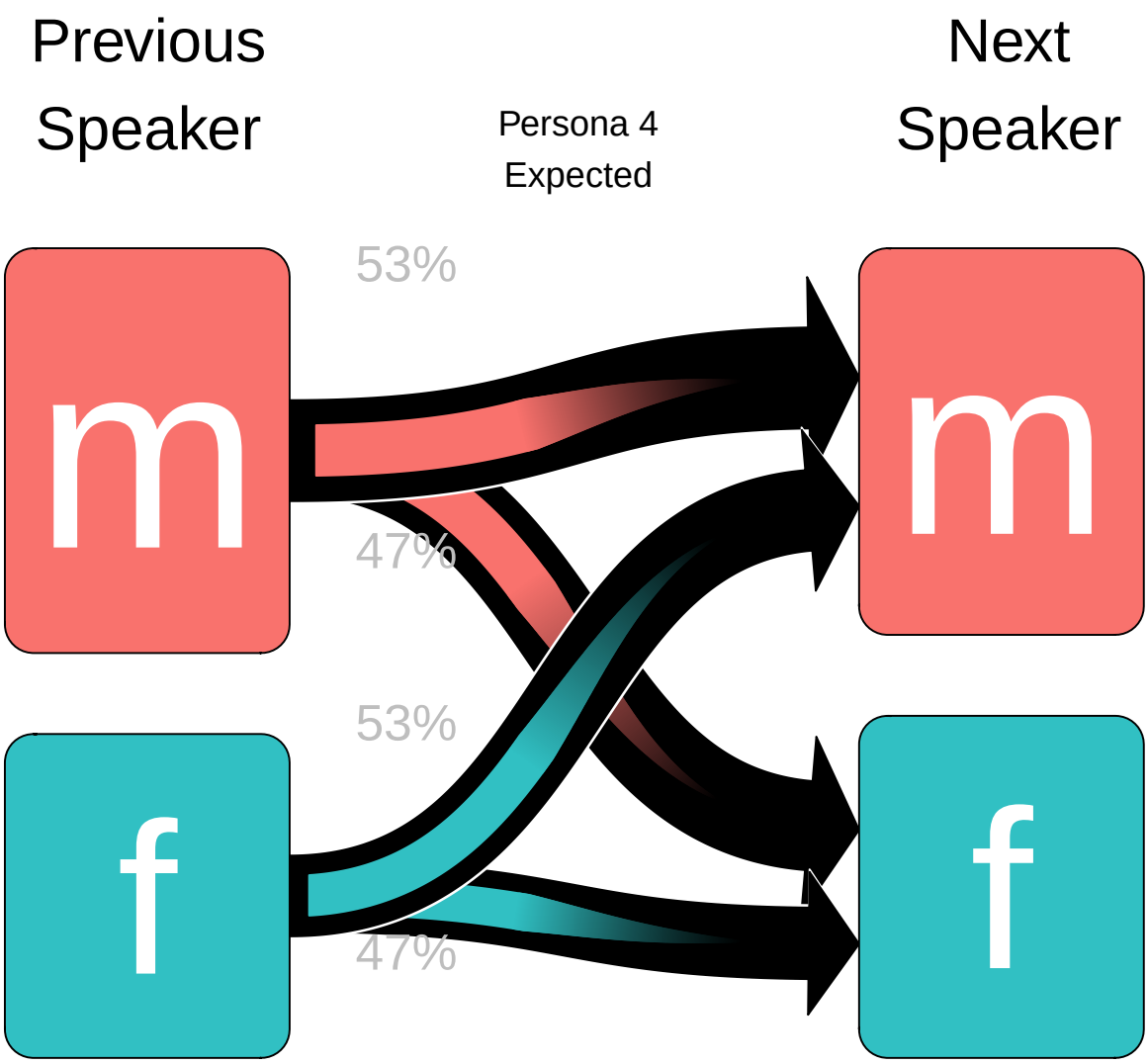

Previous Speaker
The Secret of Monkey Island
Empirical
Next Speaker
92%
8%
100%
0%
m
f
m
f

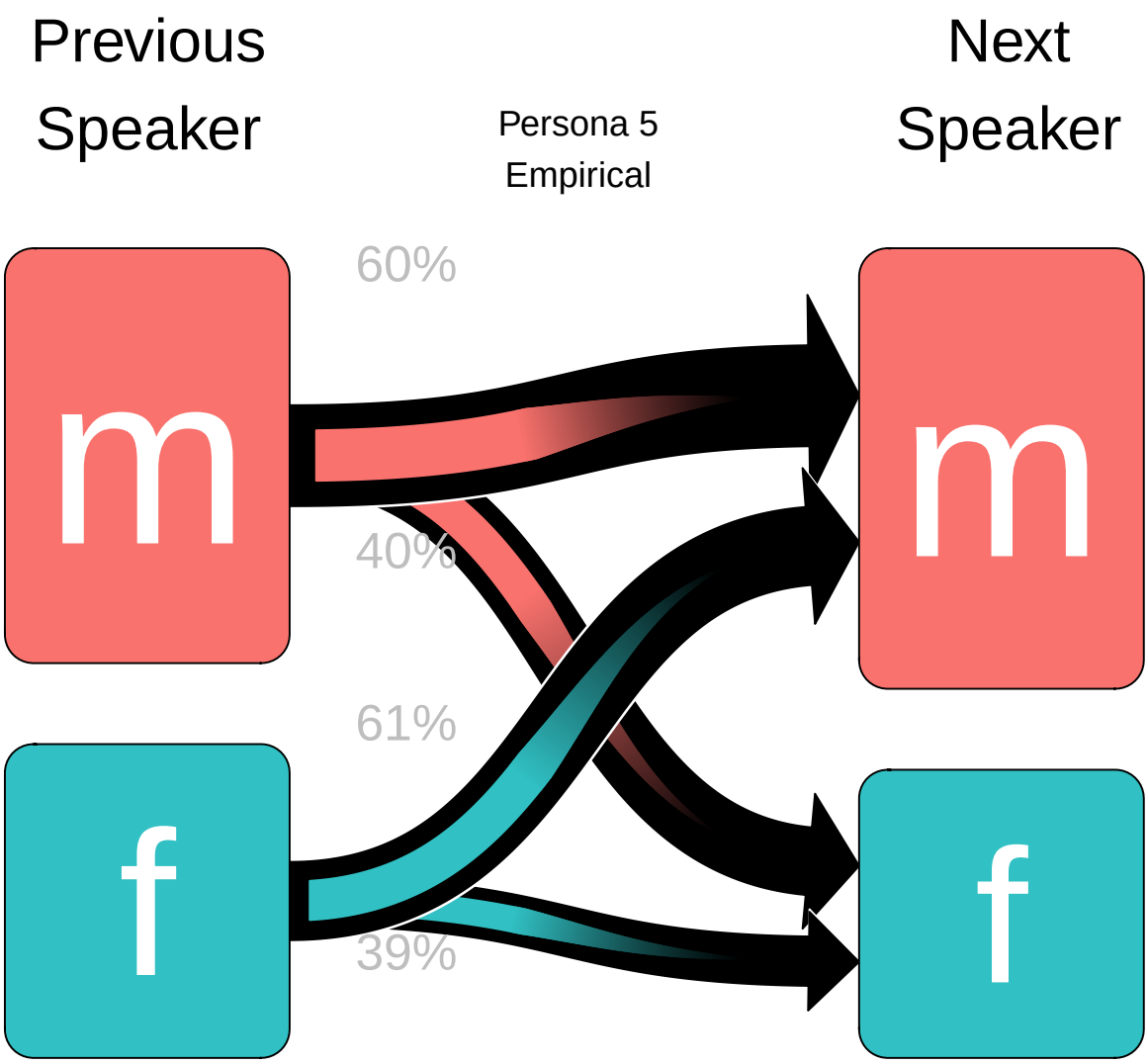

Previous Speaker
The Secret of Monkey Island
Expected
Next Speaker
93%
7%
93%
7%
m
f
m
f

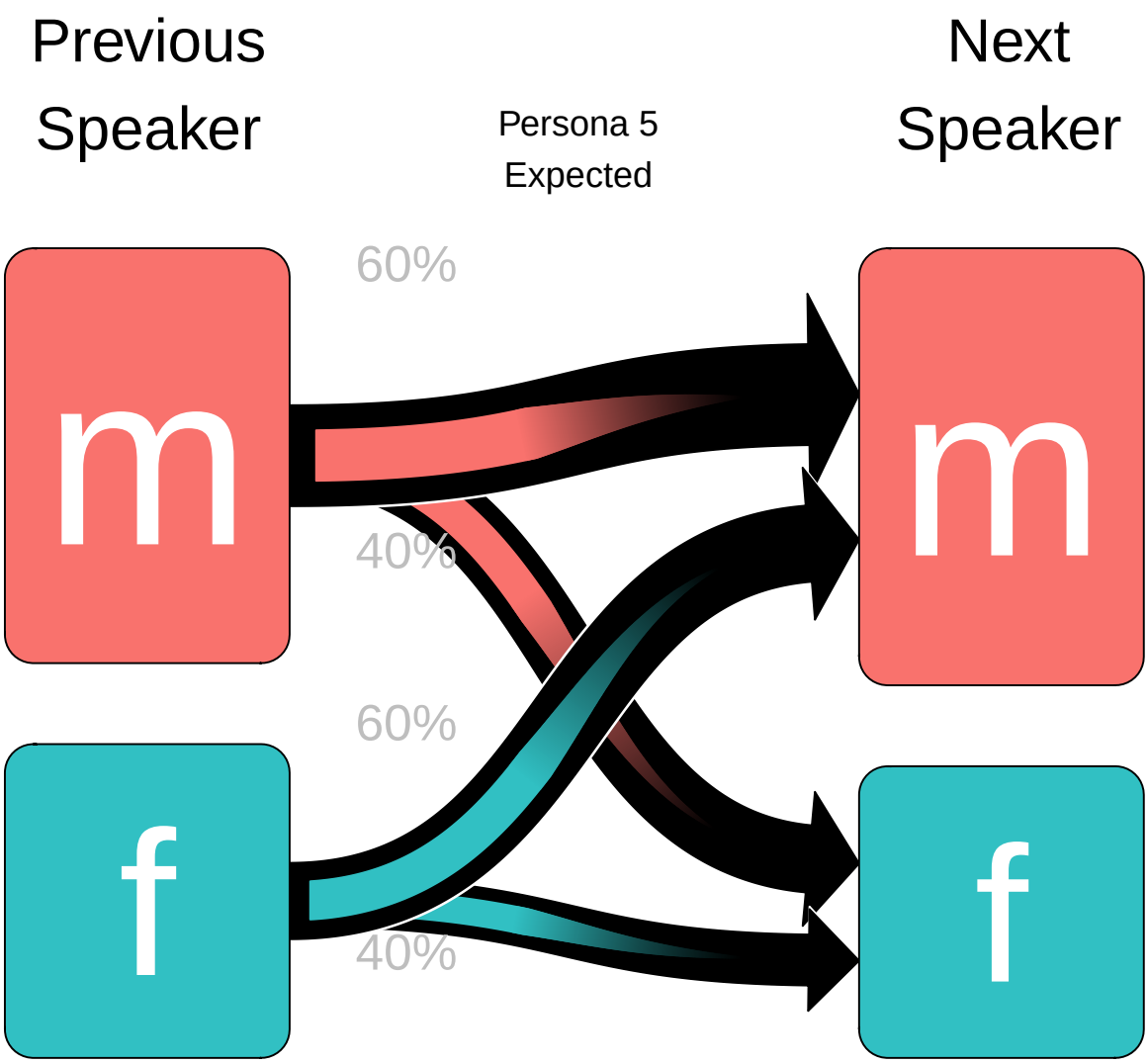

Previous Speaker
Persona 3 Empirical
Next Speaker
m
f
44%
56%
51%
49%
m
f

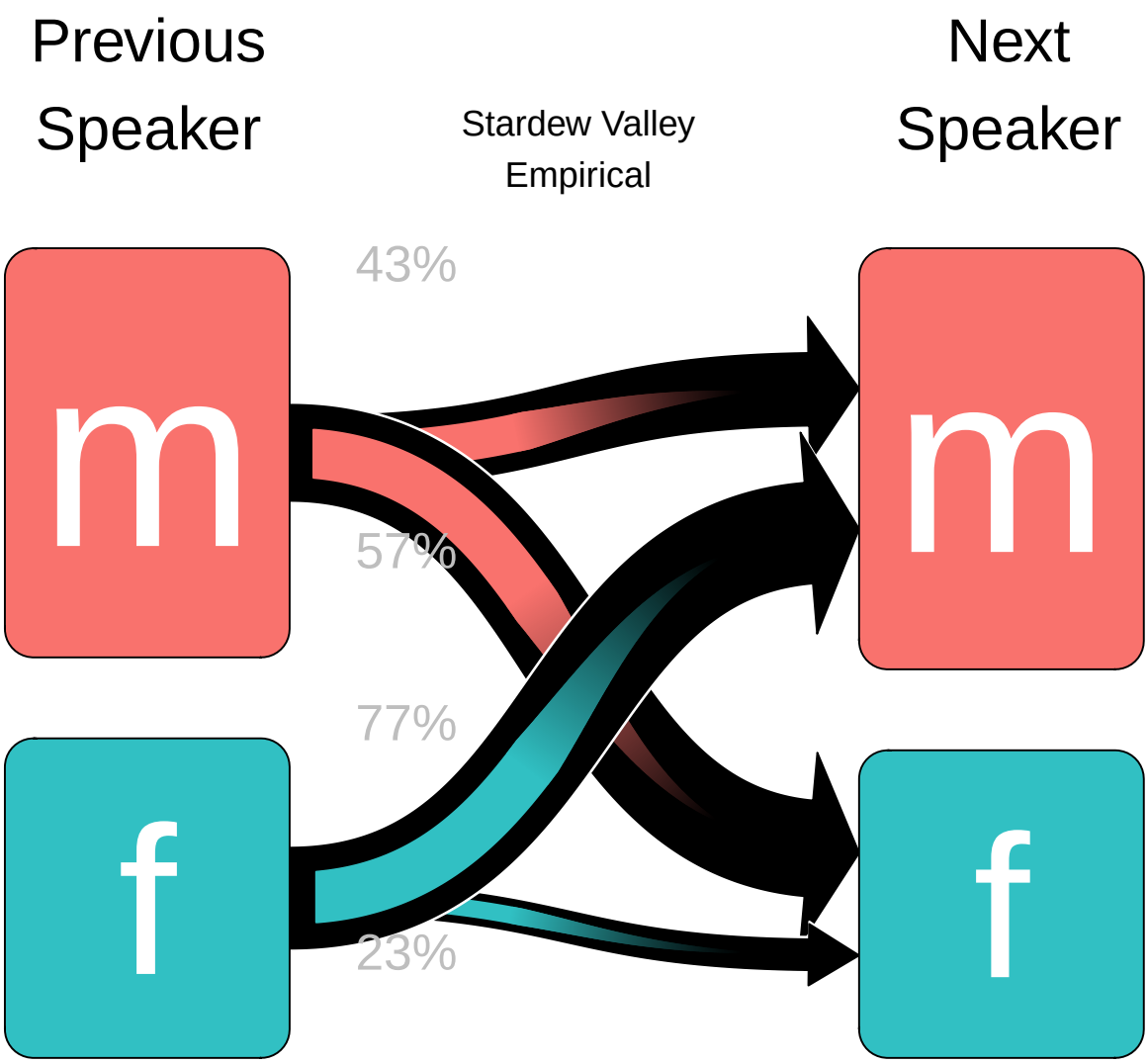

Previous Speaker
Persona 3 Expected
Next Speaker
m
f
52%
48%
52%
48%
m
f

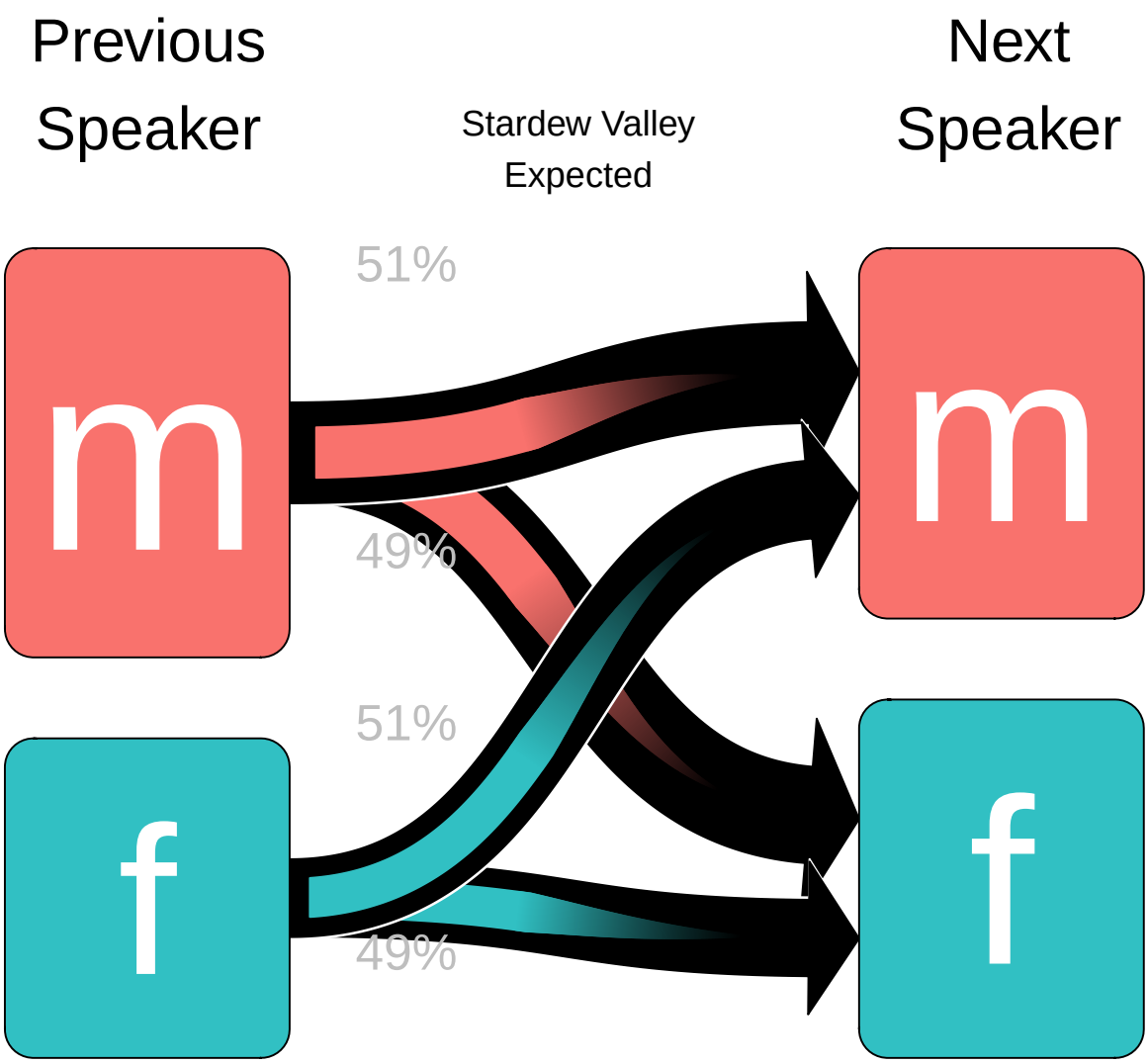

Previous Speaker
Persona 4 Empirical
Next Speaker
m
f
52%
48%
60%
40%
m
f


Previous Speaker
Persona 4 Expected
Next Speaker
m
f
53%
47%
53%
47%
m
f


Previous Speaker
Persona 5 Empirical
Next Speaker
m
f
60%
40%
61%
39%
m
f


Previous Speaker
Persona 5 Expected
Next Speaker
m
f
60%
40%
60%
40%
m
f

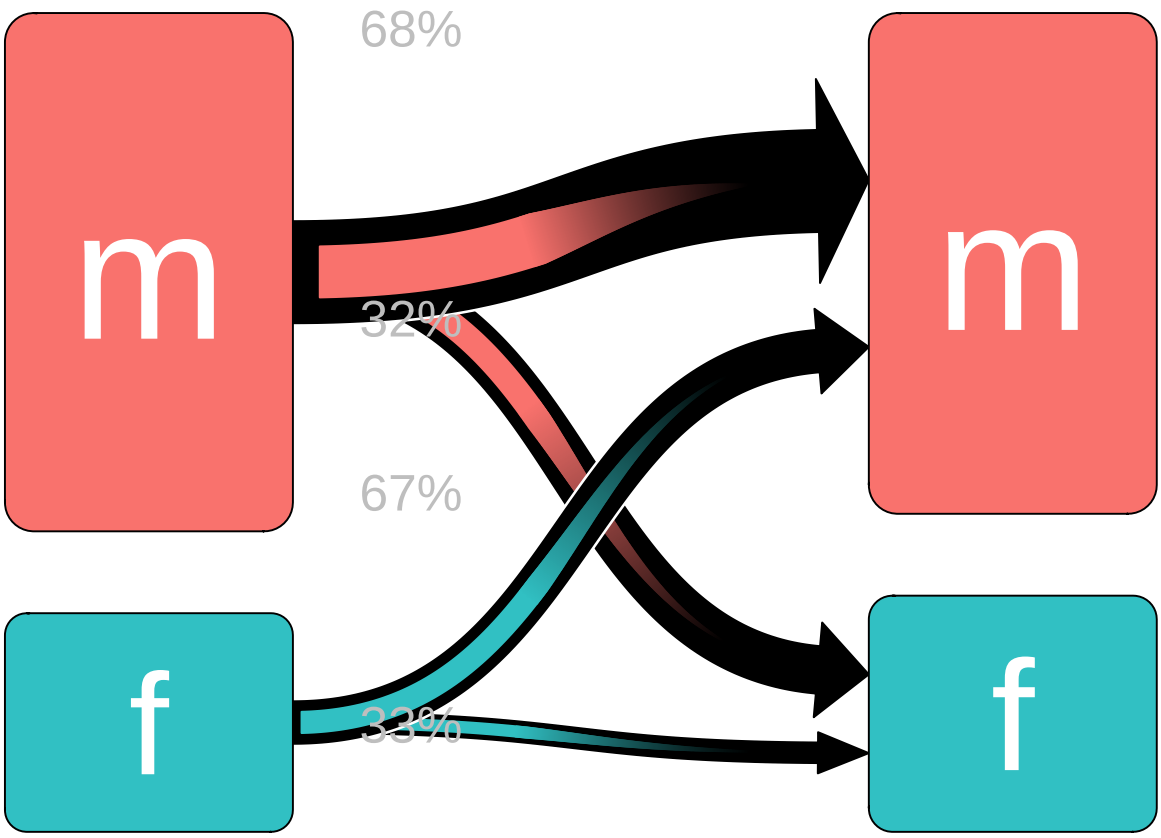

Previous Speaker
Stardew Valley Empirical
Next Speaker
m
f
43%
57%
77%
23%
m
f


Previous Speaker
Stardew Valley Expected
Next Speaker
m
f
51%
49%
51%
49%
m
f

Previous Speaker

Next Speaker

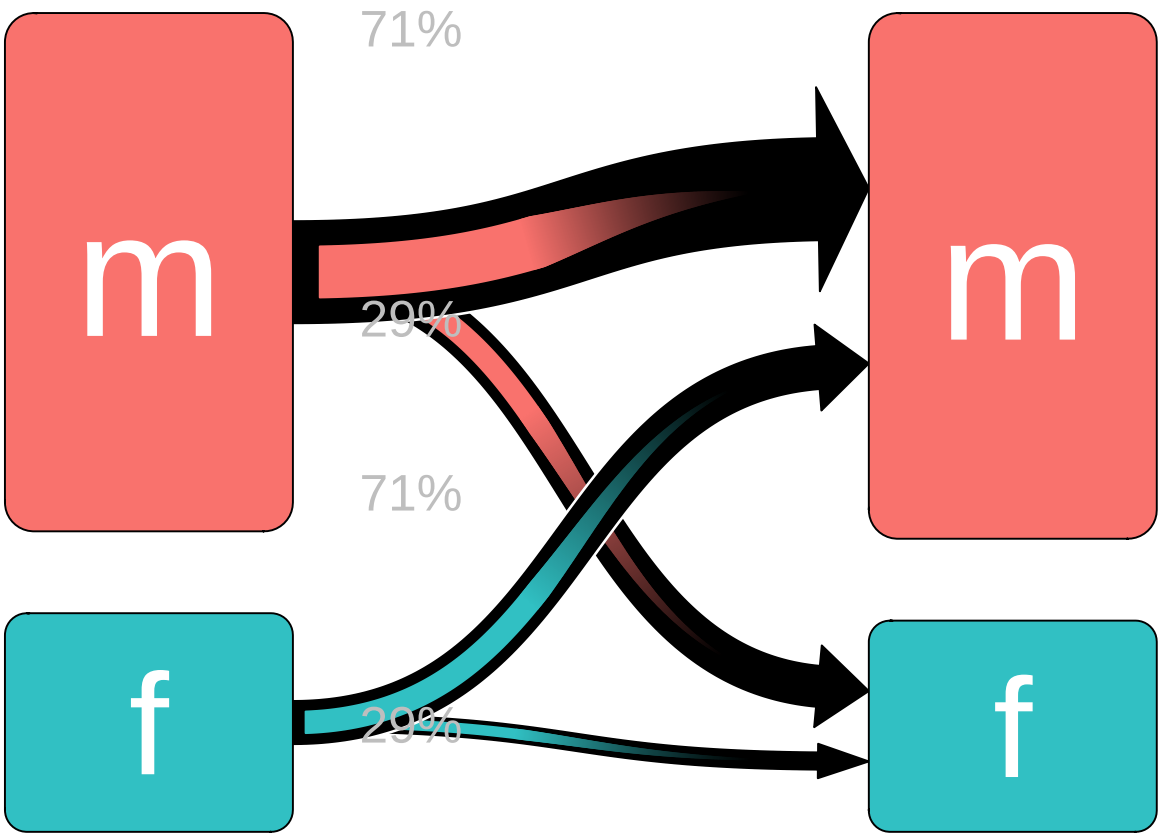
Star Wars Knights of the Old Republic
Empirical



Previous Speaker

Next Speaker

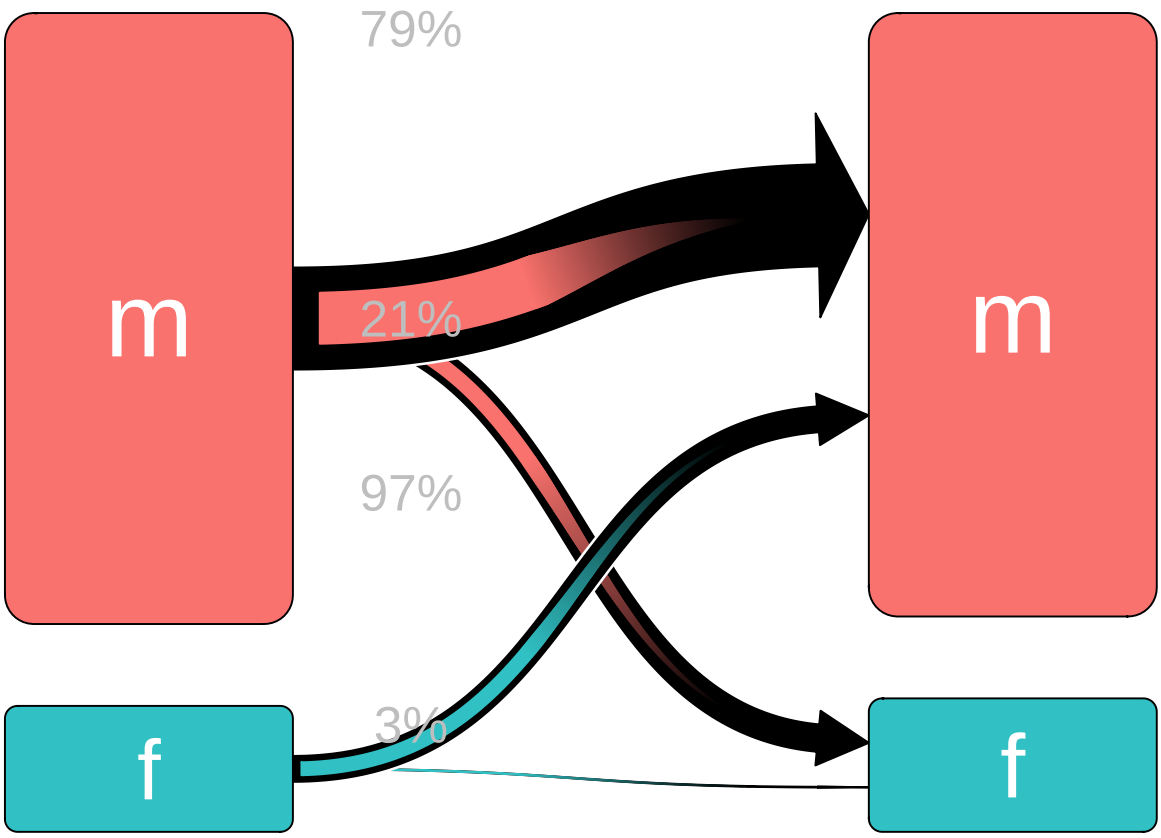
Star Wars Knights of the Old Republic Expected

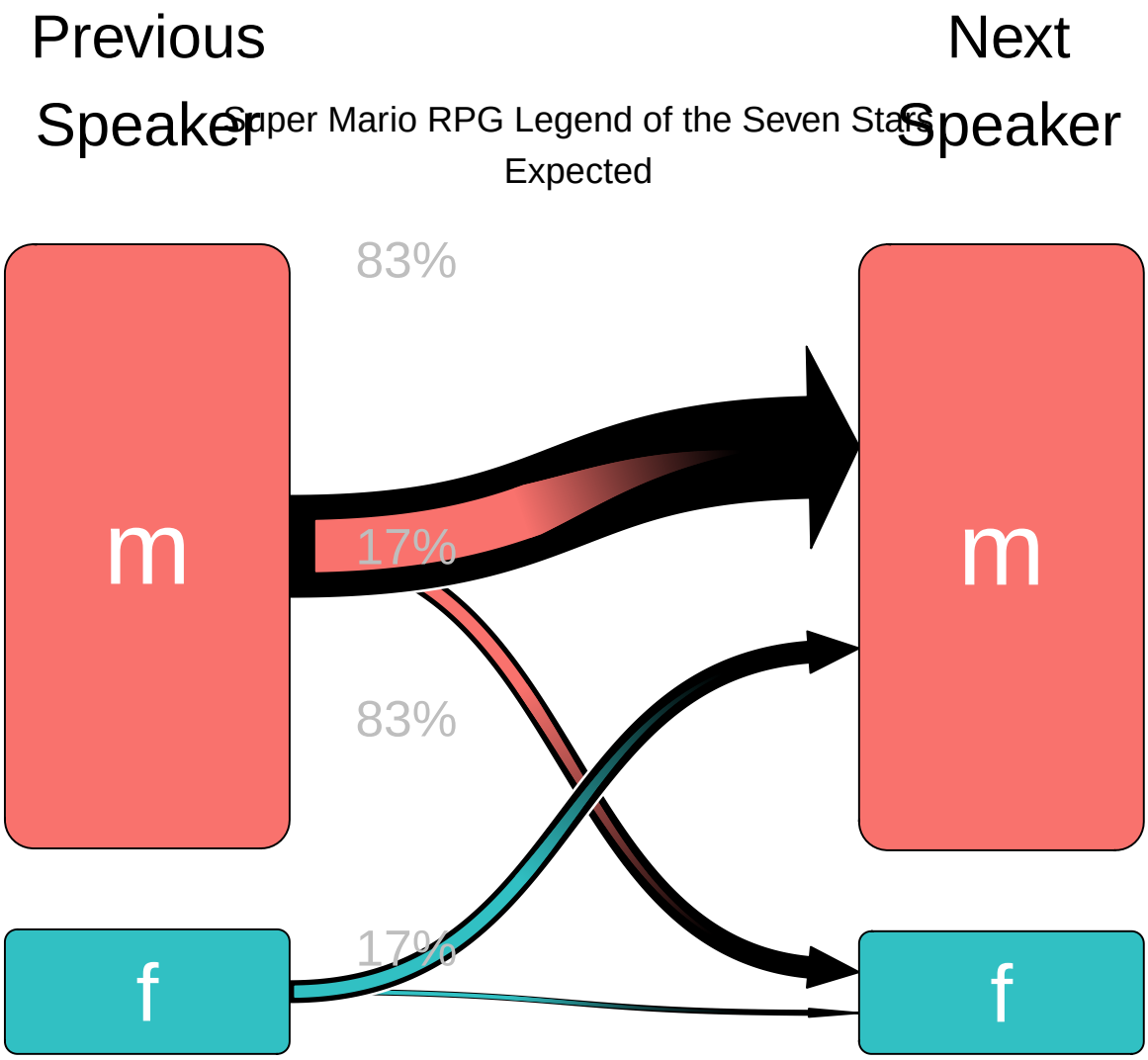


Previous Speaker

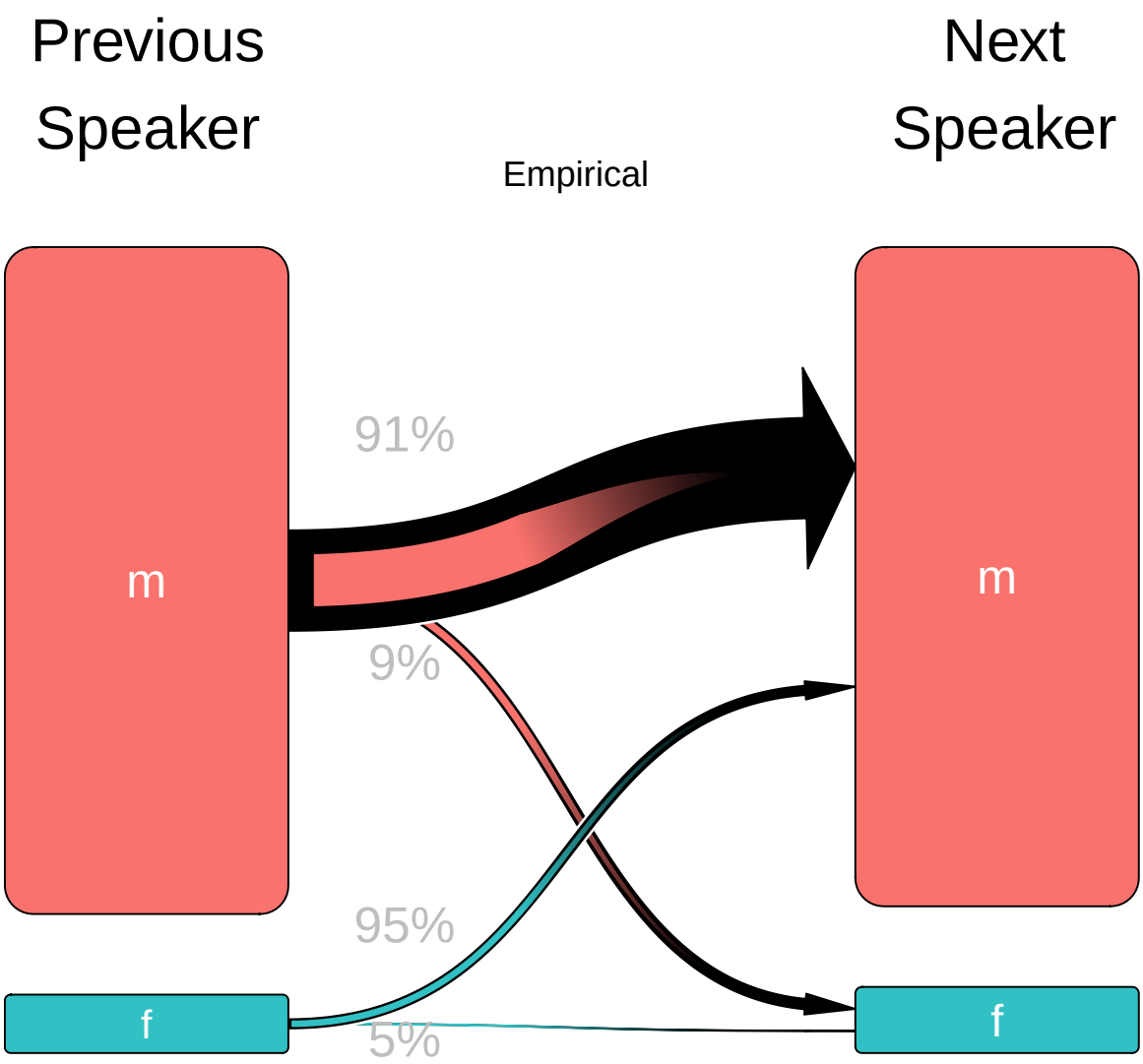
Next Speaker

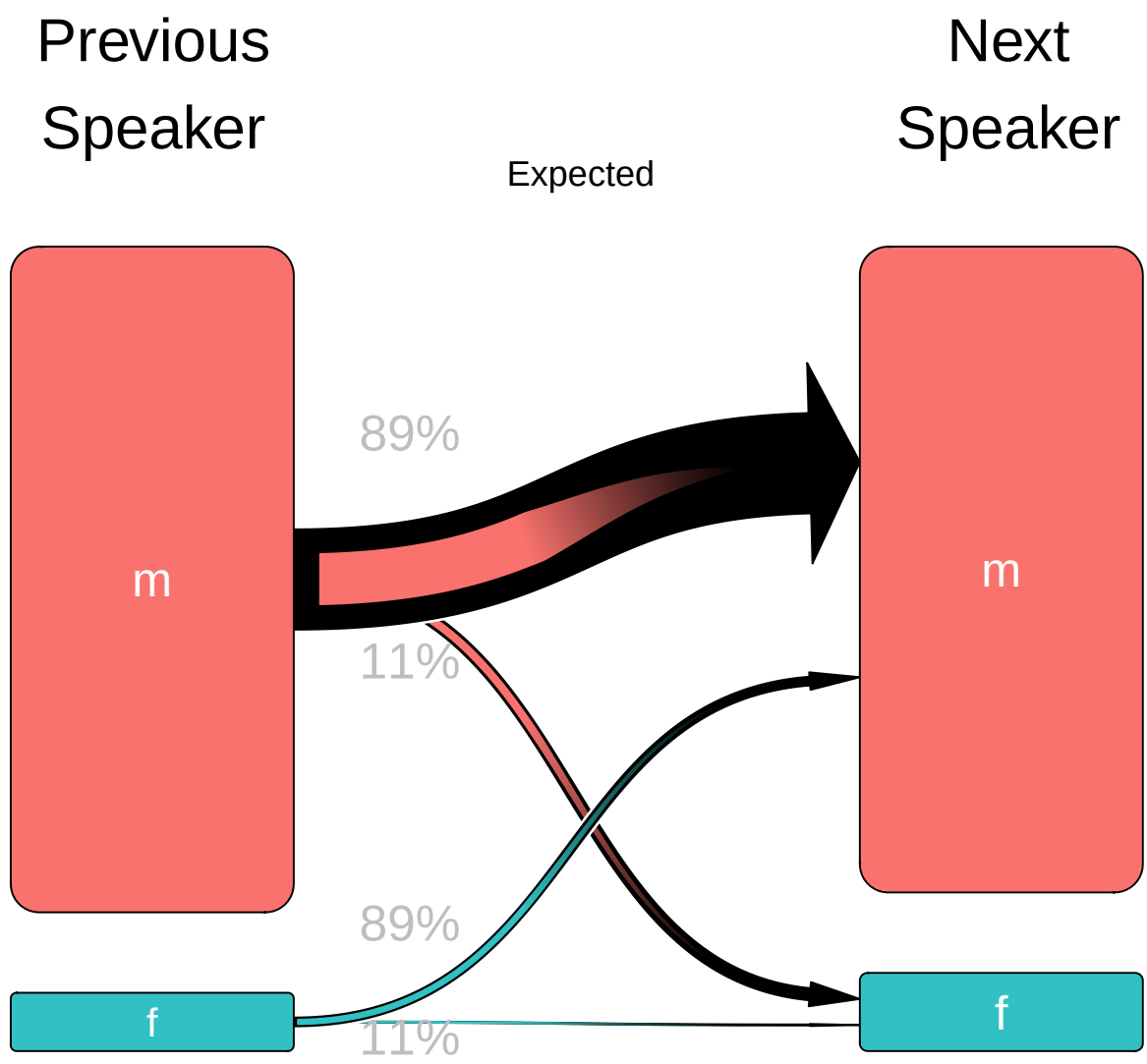
Super Mario RPG Legend of the Seven Stars
Empirical






```
makeTransitionGraphForOneGame("Final Fantasy XV", "../results/graphs/transitions/Transitions_FF15.pd
```


Previous Speaker
Next Speaker
Empirical
m
f
m
f
91%
9%
95%
5%




```
## pdf
##   2
```

Write stats for paper:

```
numGames = length(unique(permutationResults$folder))
cat(numGames, file="../results/latexStats/transitions_NumGames.tex")

cat(round(100*trueTransitions.AllGames['m','m']),
    file= "../results/latexStats/transitions_MMPer.tex")
MM.moreThanExpected = trueTransitions.AllGames['m','m'] -
  permutedTransitionStats.AllGames['m2m.mean']
cat(round(100*MM.moreThanExpected),
    file= "../results/latexStats/transitions_MMMoreThanExpected.tex")
MMp = paste("=",permutedTransitionStats.AllGames['m2m.p'])
if(MMp <= 0.001){
  MMp = "< 0.001"
}
MMStats = paste0(
  "z = ", round(permutedTransitionStats.AllGames['m2m.z'],2),
  ", p = ", MMp)
```

```
cat(MMStats, file = "../results/latexStats/transitions_MMStats.tex")
MMNumGamesNotBiased = sum(permutationResults$m2m.p>0.05)
cat(MMNumGamesNotBiased, file = "../results/latexStats/transitions_MMNumGamesNotBiased.tex")

# For F-F transitions
cat(round(100*trueTransitions.AllGames['f','f']),
    file= "../results/latexStats/transitions_FFPer.tex")
FF.lessThanExpected = permutedTransitionStats.AllGames['f2f.mean'] -
  trueTransitions.AllGames['f','f']
cat(round(100*FF.lessThanExpected),
    file= "../results/latexStats/transitions_FFLessThanExpected.tex")
FFp = paste("=",permutedTransitionStats.AllGames['f2f.p'])
if(FFp <= 0.001){
  FFp = "< 0.001"
}
FFStats = paste0(
  "z = ", round(permutedTransitionStats.AllGames['f2f.z'],2),
  ", p ", FFp)
cat(FFStats, file = "../results/latexStats/transitions_FFStats.tex")
FFNumGamesNotBiased = sum(permutationResults$f2f.p>0.05)
cat(FFNumGamesNotBiased,
    file = "../results/latexStats/transitions_FFNumGamesNotBiased.tex")

FFp = paste("=",permutationResults[permutationResults$game=="Final Fantasy X-2",]$f2f.p)
if(FFp <= 0.001){
  FFp = "< 0.001"
}
FF.FFX2Stats = paste0(
  "z = ", round(permutedTransitionStats.AllGames['f2f.z'],2),
  ", p ", FFp)
cat(FF.FFX2Stats,
    file = "../results/latexStats/transitions_FF_FFX2Stats.tex")

FFp = paste("=",permutationResults[permutationResults$game=="Final Fantasy X-2",]$m2m.p)
if(FFp <= 0.001){
  FFp = "< 0.001"
}
MM.FFX2Stats = paste0(
  "z = ", round(permutedTransitionStats.AllGames['m2m.z'],2),
  ", p ", FFp)
cat(MM.FFX2Stats,
    file = "../results/latexStats/transitions_MM_FFX2Stats.tex")
```

References

Bechdel, Allison. Dykes to Watch Out For. Firebrand Books (October 1, 1986).

Agarwal, A., Zheng, J., Kamath, S., Balasubramanian, S., & Dey, S. A. (2015). Key female characters in film have more to talk about besides men: Automating the bechdel test. In Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 830-840).

Weng, C. Y., Chu, W. T., & Wu, J. L. (2009). Rolenet: Movie analysis from the perspective of social networks. IEEE Transactions on Multimedia, 11(2), 256-271.