

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 arc. 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 Bechdel 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 permutation 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 linguistic 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

```
}

## [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 Bechdel 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 =
  getPermutatedStats(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   0.5678514
## m2m.z      10.4406596
## m2m.p      0.0001000
## f2m.mean   0.5680084
## f2m.z      9.1144318
## f2m.p      0.0001000
## m2f.mean   0.4321486
## m2f.z      -10.4406596
## m2f.p      0.0001000
## f2f.mean   0.4319916
## f2f.z      -9.1144318
## f2f.p      0.0001000

```

Stats for each game:

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

```

    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

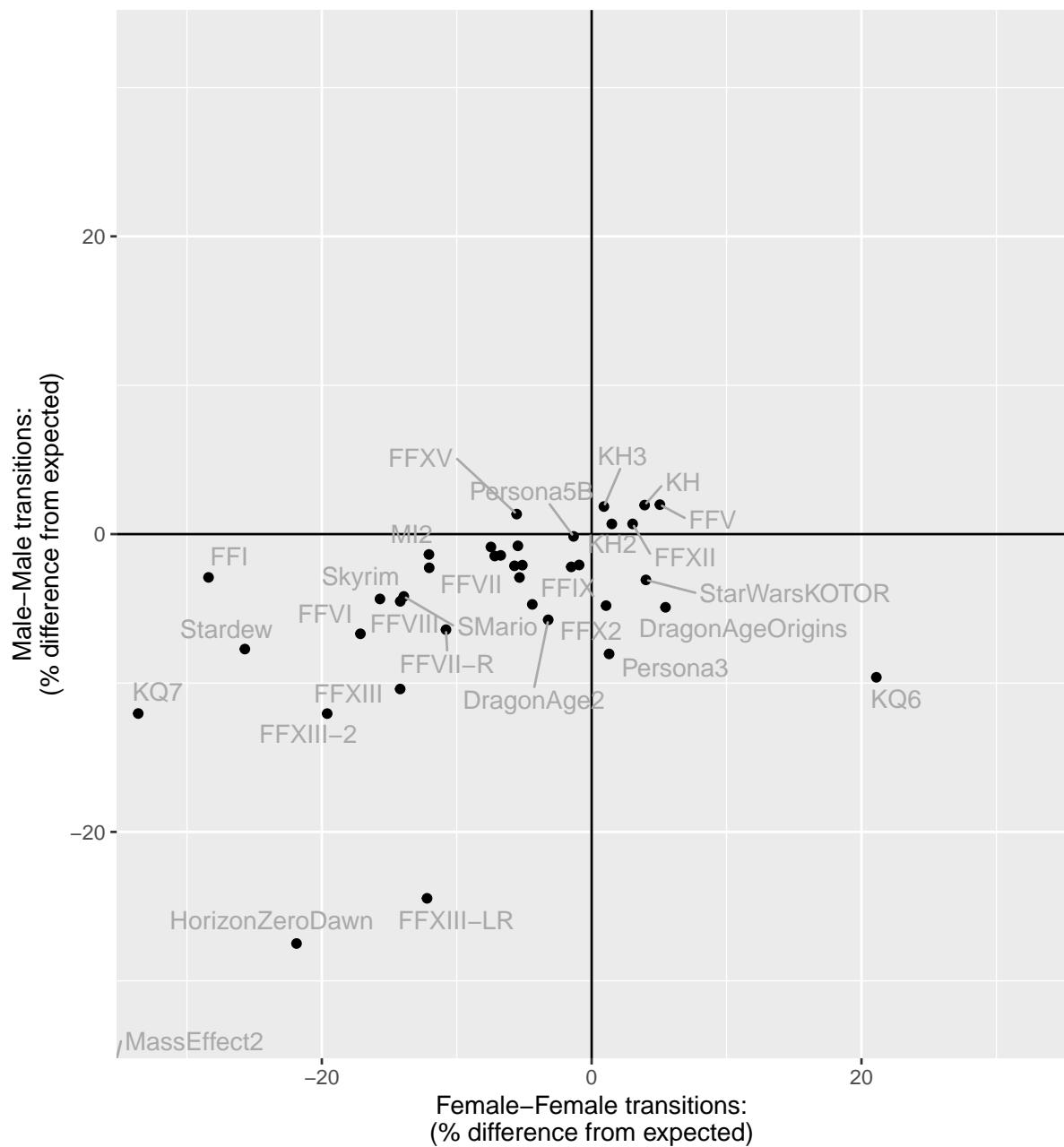
Write the data:
write.csv(permutationResults, "../results/transitionsPermutationTest.csv", row.names = F)

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(permuationResults, 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) +

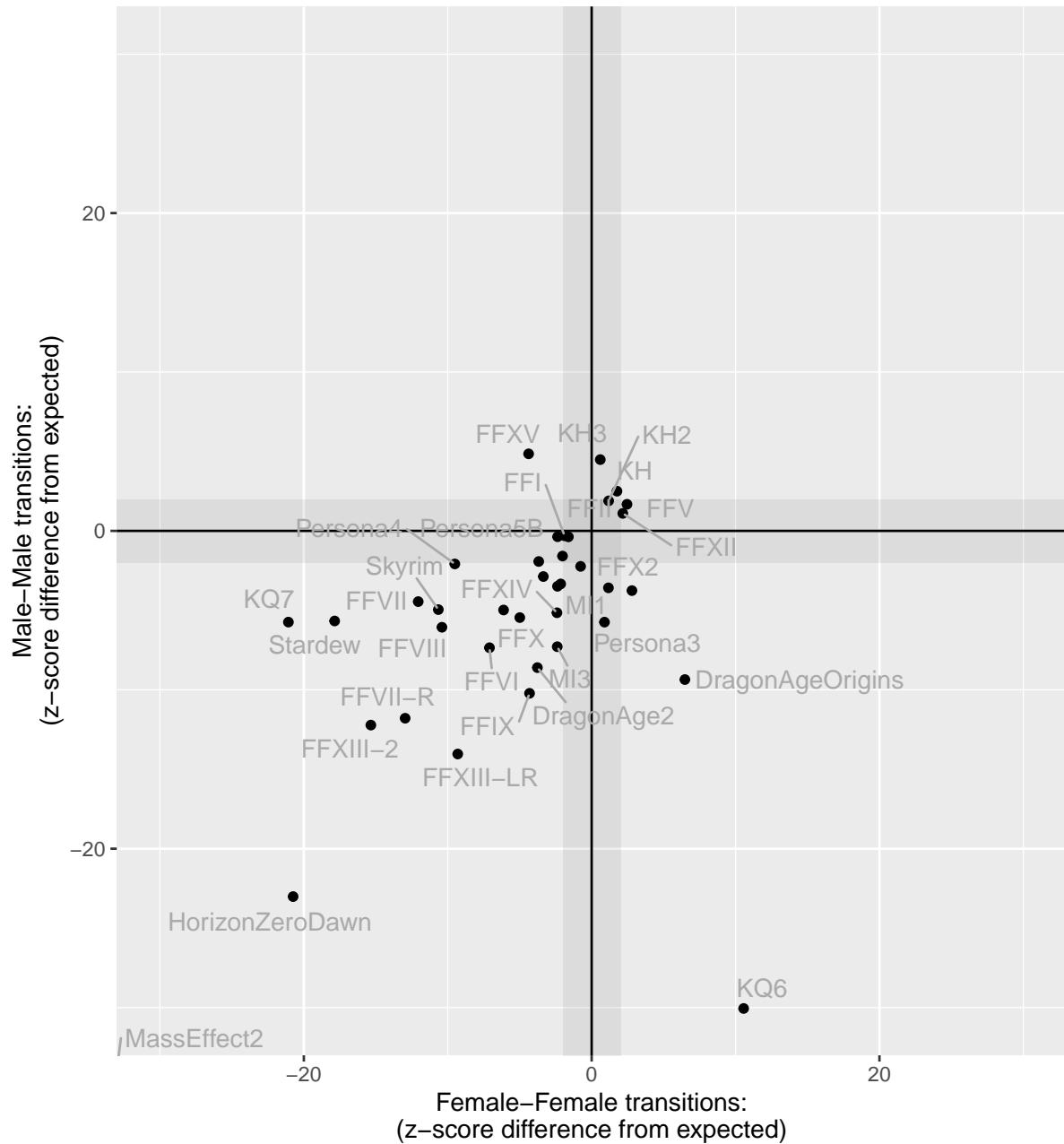
```

```

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(permResults,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()

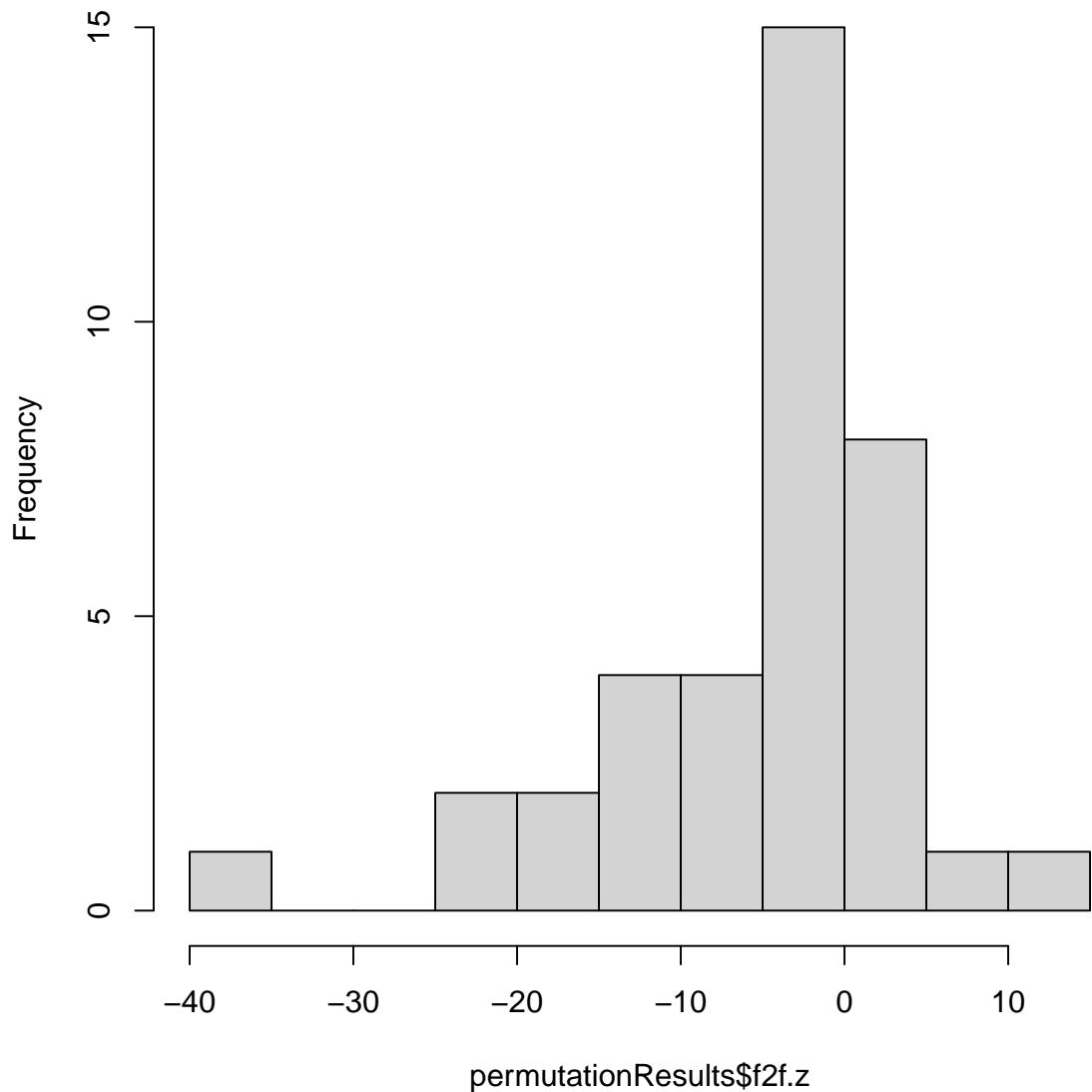
## pdf
## 2

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

hist(permuationResults$f2f.z)

```

Histogram of permutationResults\$f2f.z

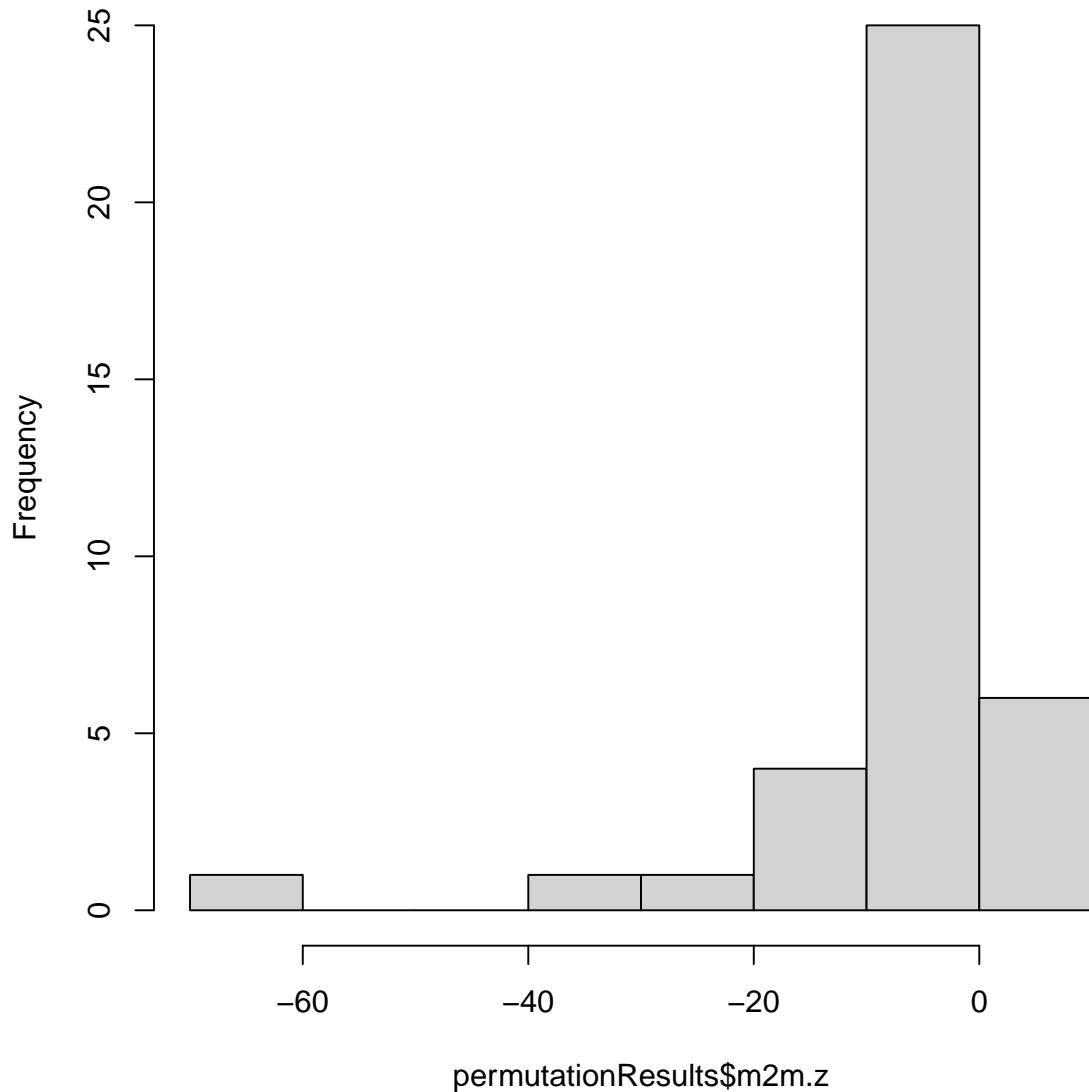


```
mean(permutationResults$f2f.z)
```

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

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

Histogram of permutationResults\$m2m.z

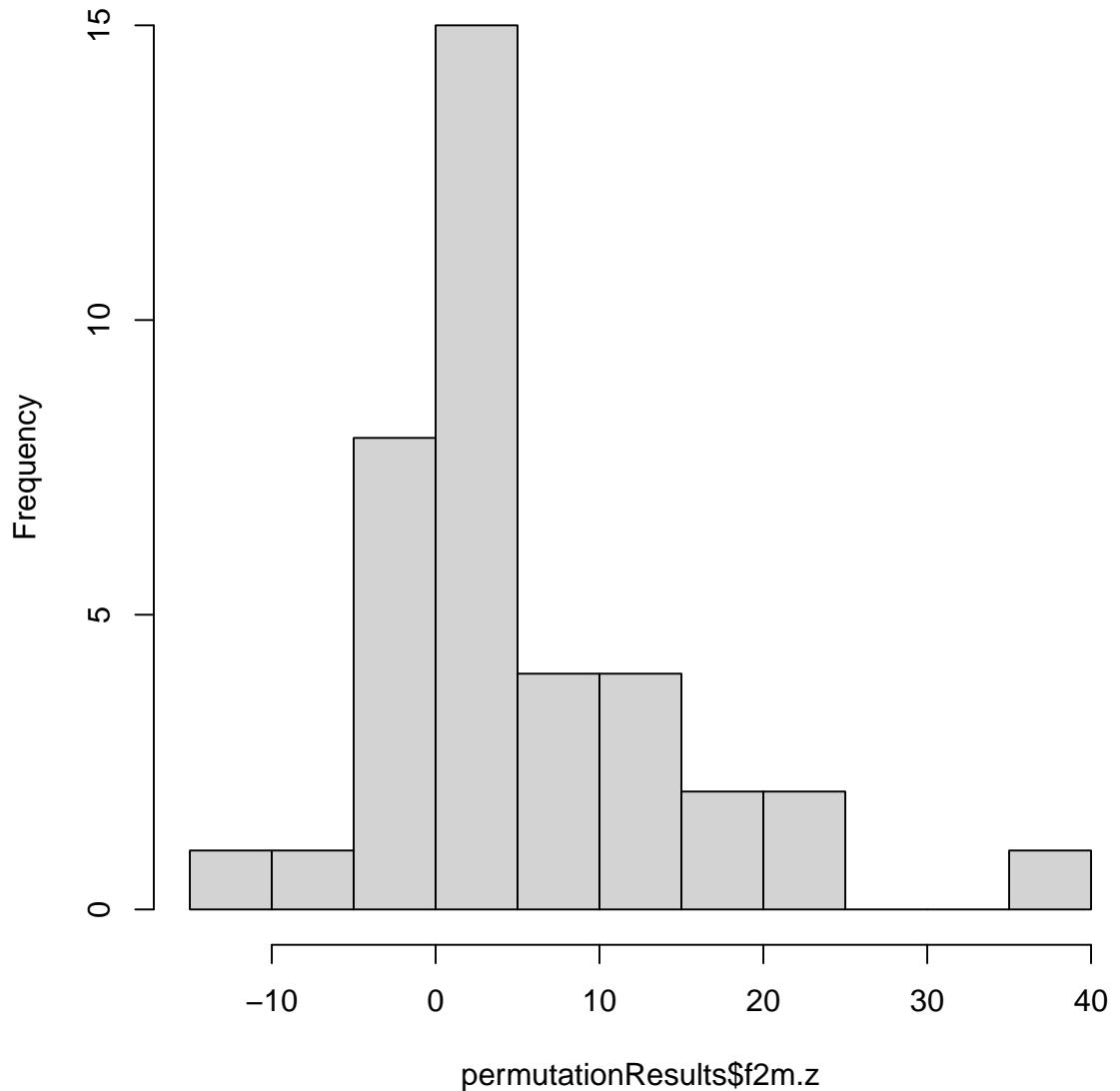


```
mean(permutationResults$m2m.z)
```

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

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

Histogram of permutationResults\$f2m.z

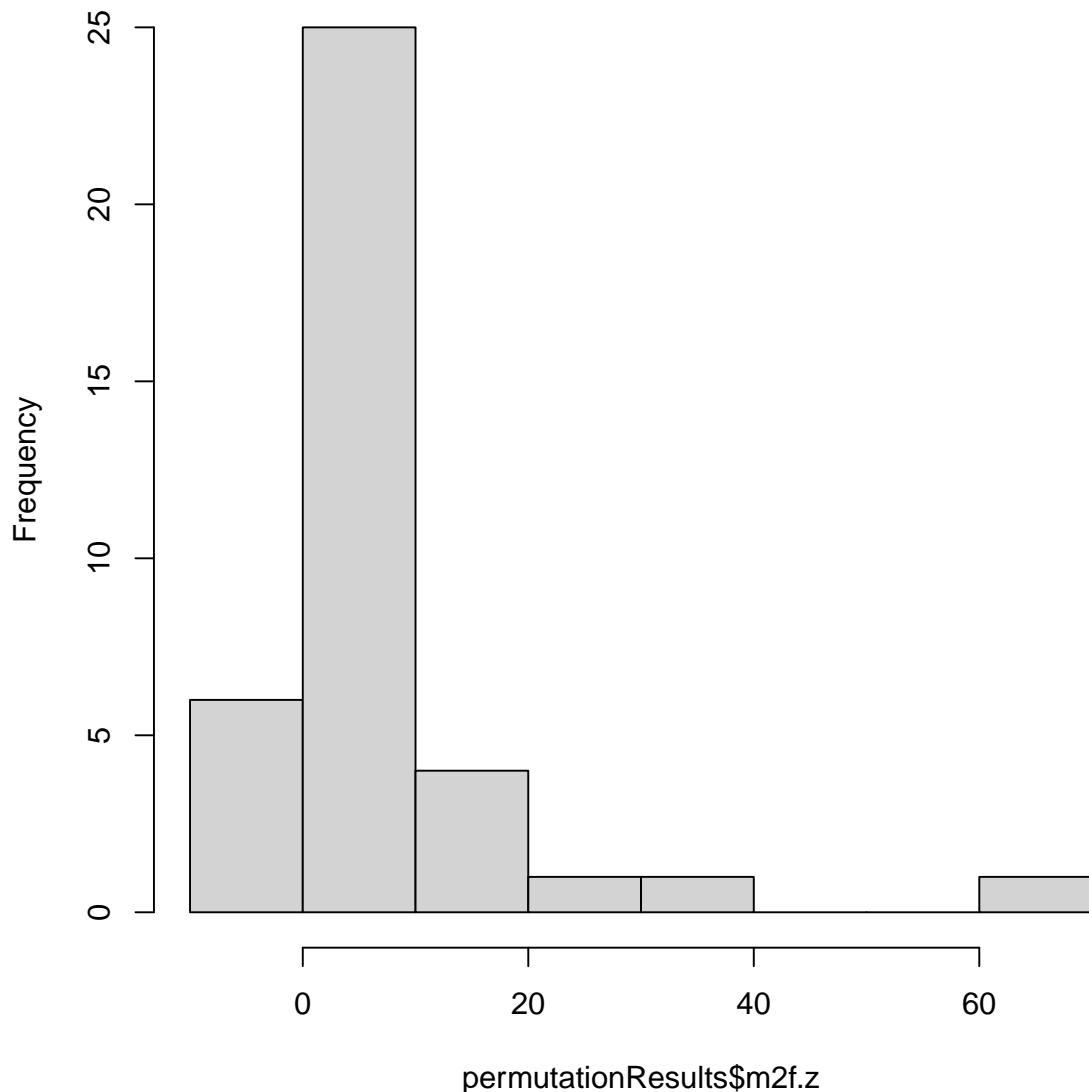


```
mean(permutationResults$f2m.z)
```

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

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

Histogram of 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(permuteResults$f2f.z < permuteResults$m2m.z))

##
## Exact binomial test
##
## data: table(permuteResults$f2f.z < permuteResults$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(permuteResults$m2m.z < 0 & permuteResults$m2m.p<0.05,
      permuteResults$f2f.z < 0 & permuteResults$f2f.p<0.05)

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

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

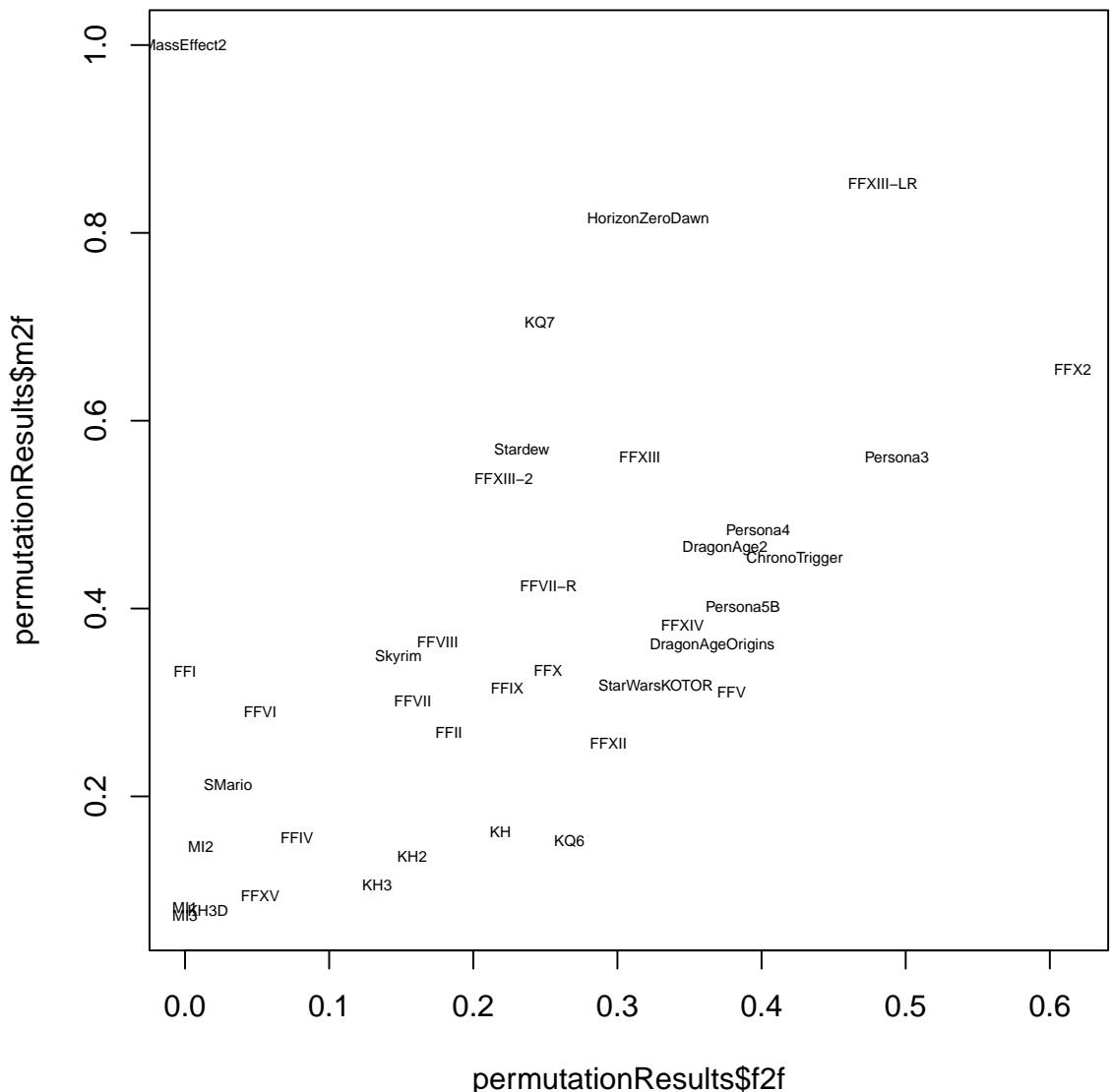
```

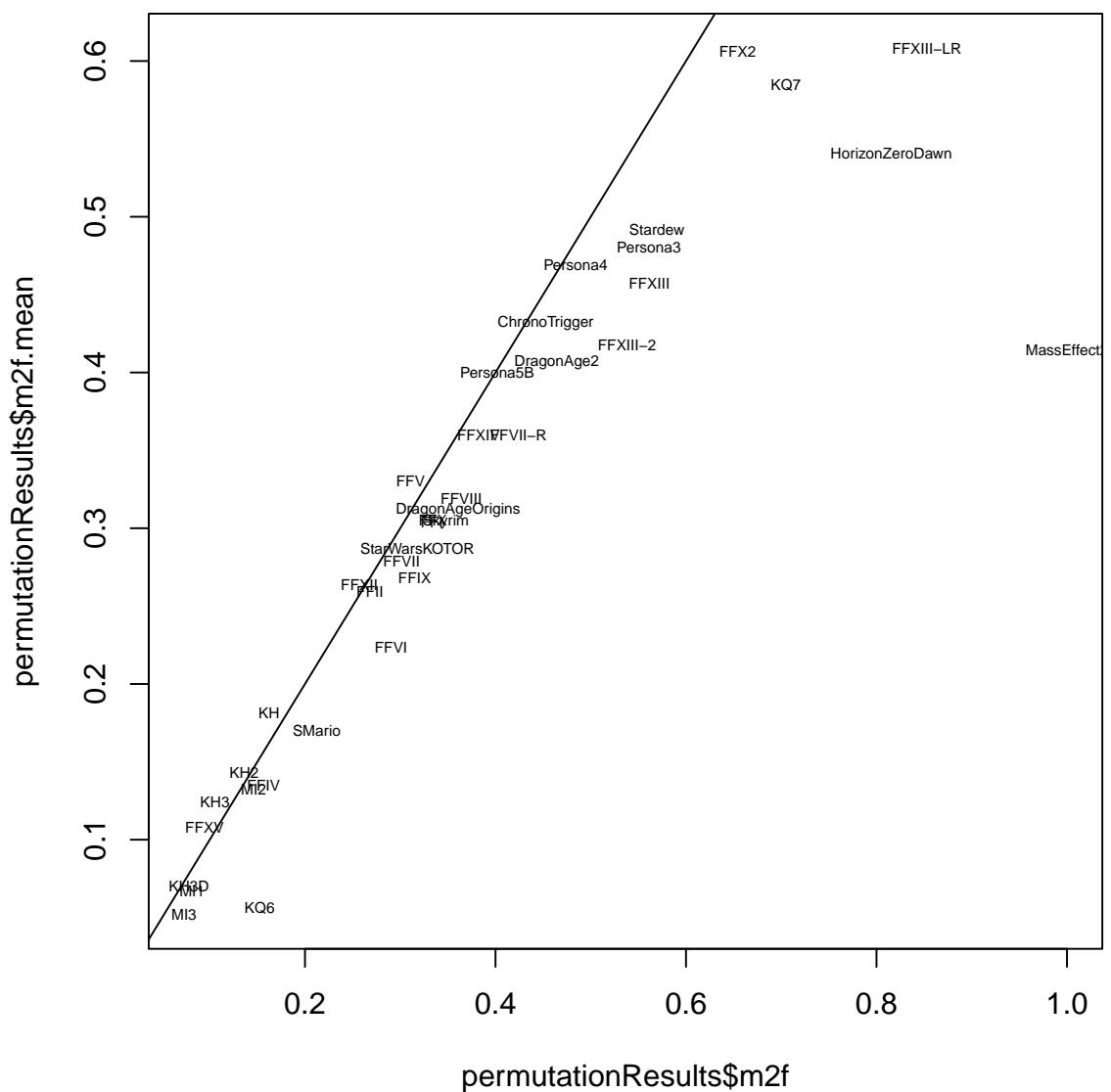
The relationship between the different transition types is correlated, but there is some variation:

```

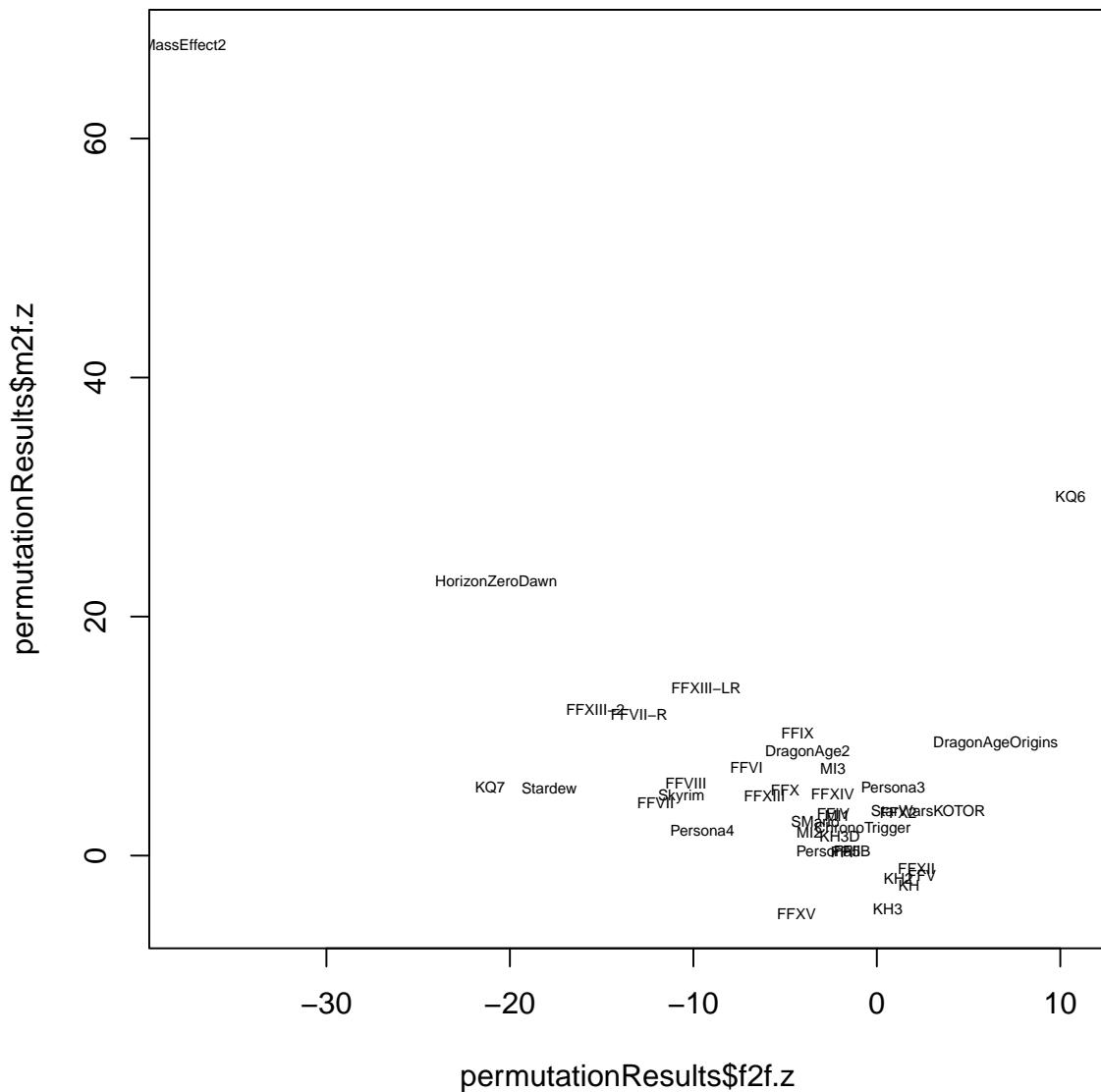
plot(permuationResults$f2f,permuationResults$m2f,col=NA)
text(permuationResults$f2f,permuationResults$m2f,permuationResults$shortName2,cex=0.5)

```

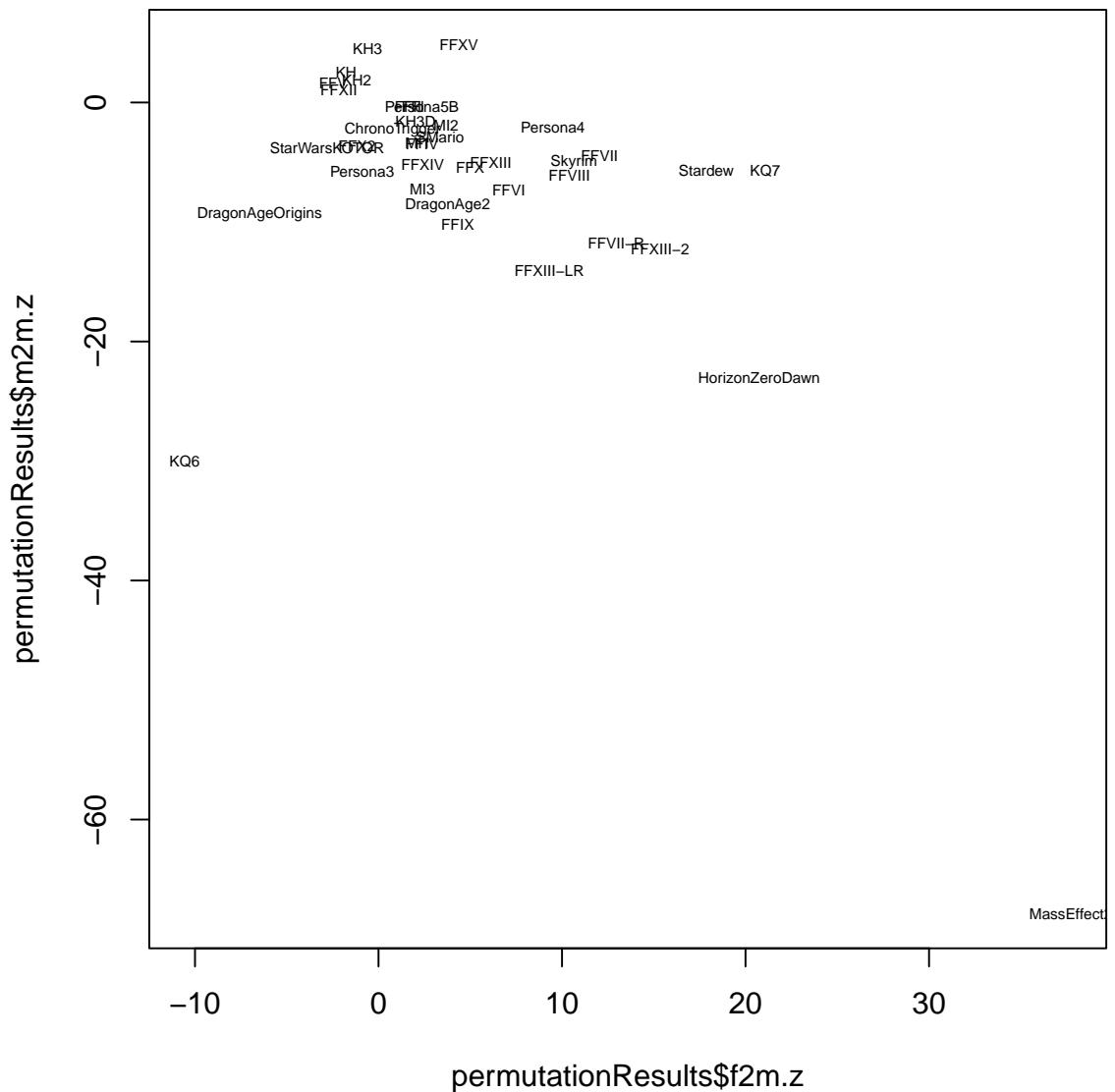




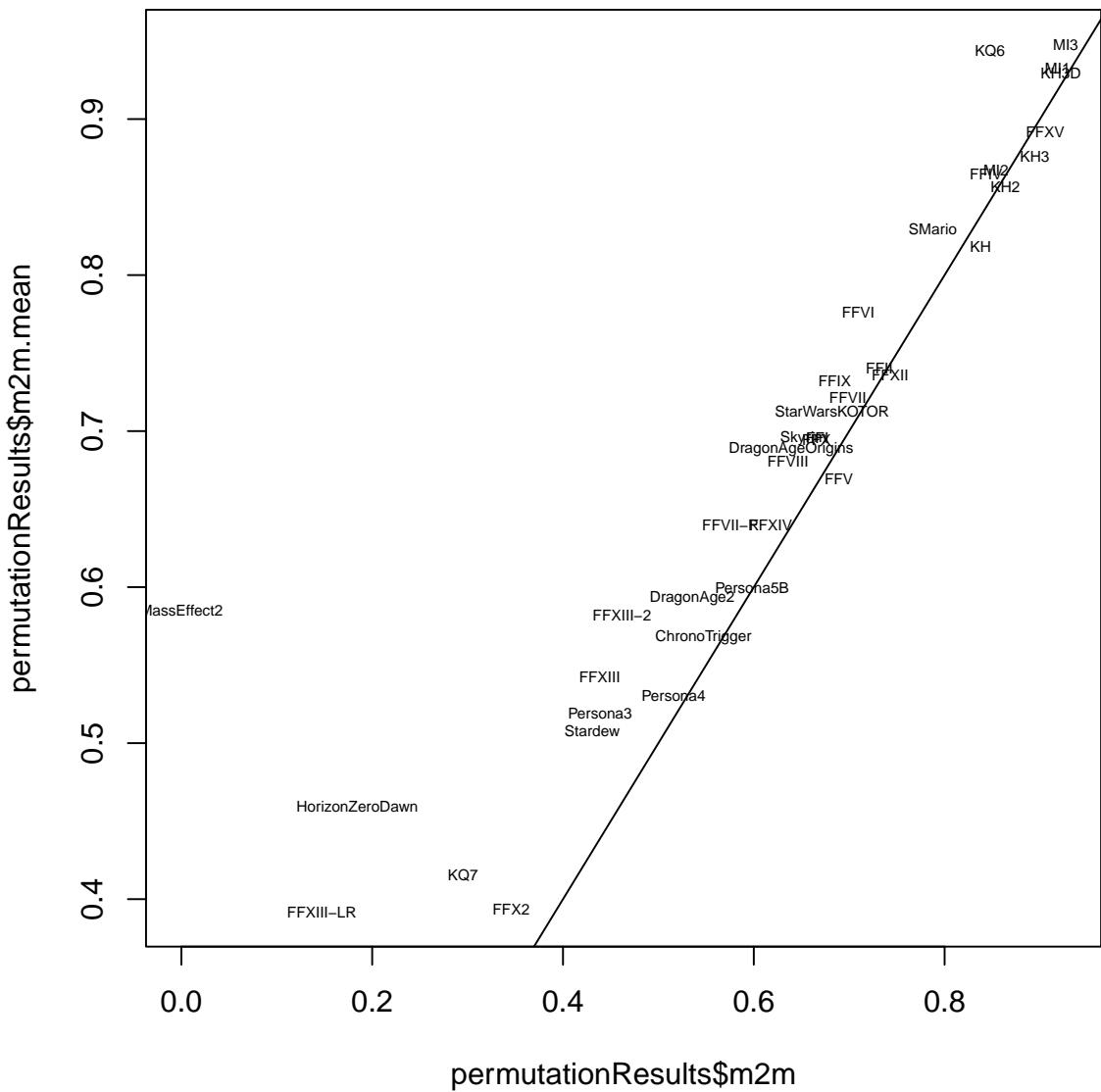
```
plot(permutationResults$f2f.z,permutationResults$m2f.z,col=NA)
text(permutationResults$f2f.z,permutationResults$m2f.z,permutationResults$shortName2,cex=0.5)
```



```
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")))

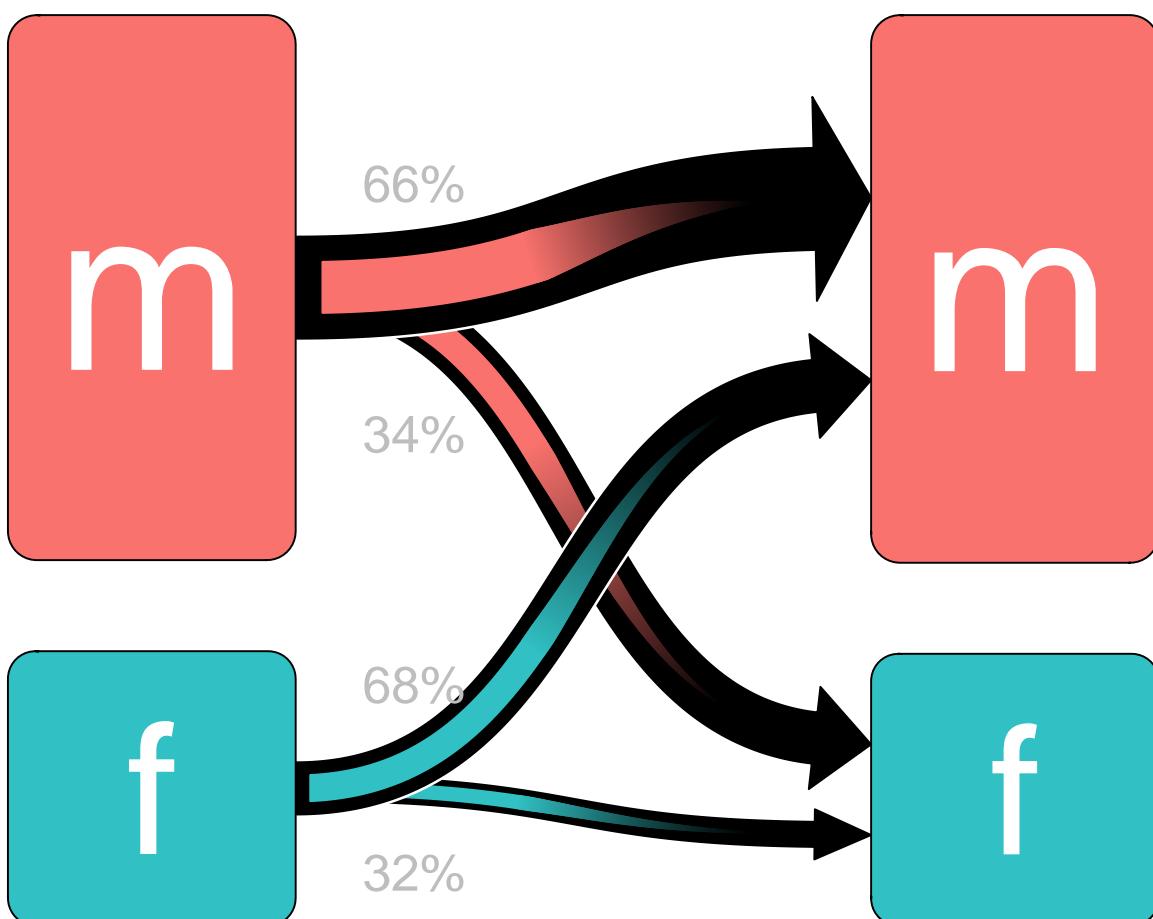
```

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"))
```

Previous Speaker Next Speaker



```
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()
```

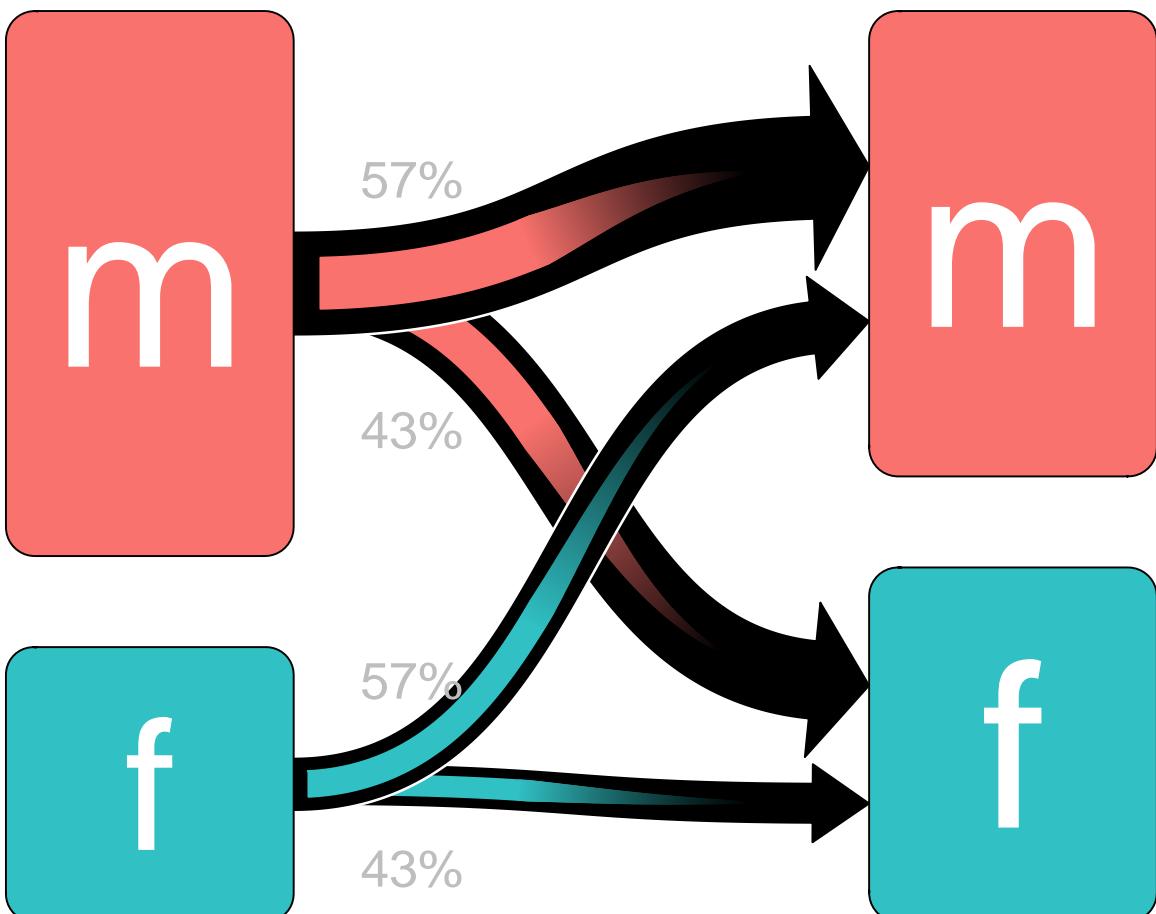
```
## pdf  
## 2
```

Expected proportions under permutation for all games:

```
expectedProp = matrix(c(permutedTransitionStats.AllGames['m2m.mean'], permutedTransitionStats.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"))
```

Previous Speaker

Next Speaker



```
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"))  
}
```

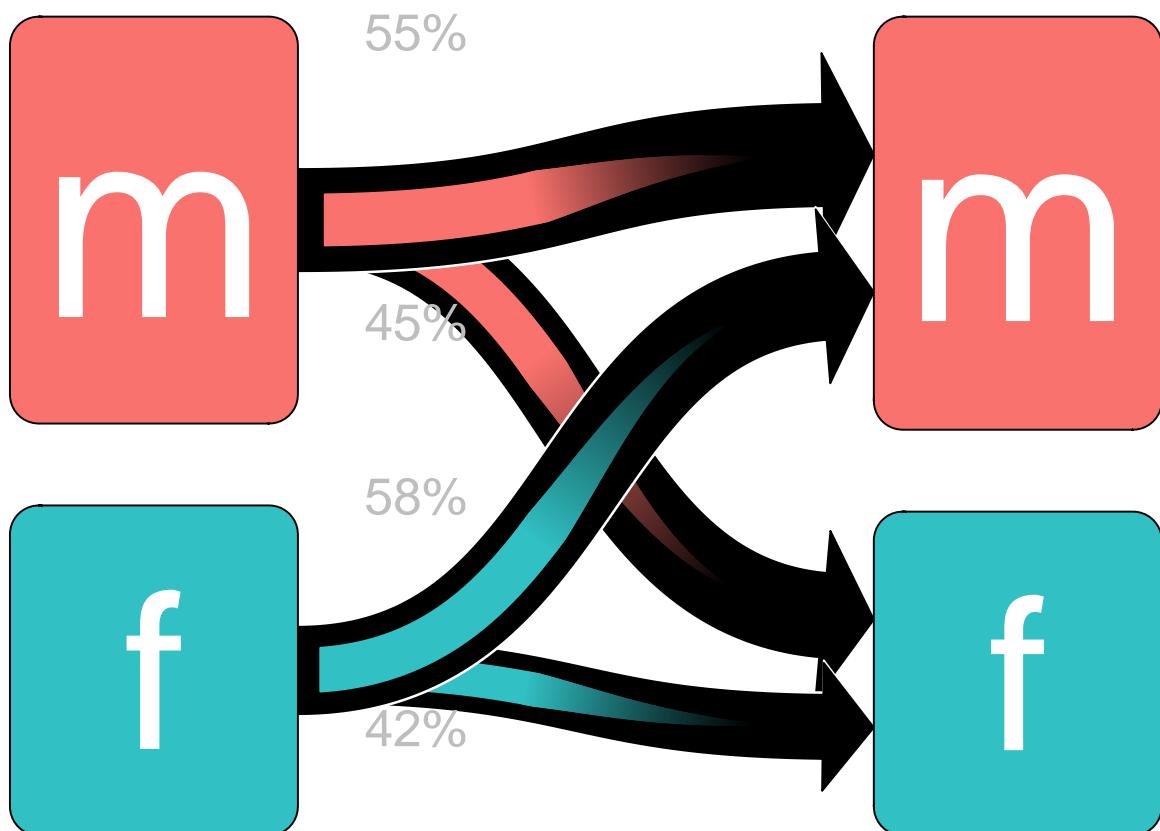
Apply to each game separately:

```
for(folder in unique(permuationResults$folder)){
  gameName = permuationResults[permuationResults$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
Empirical

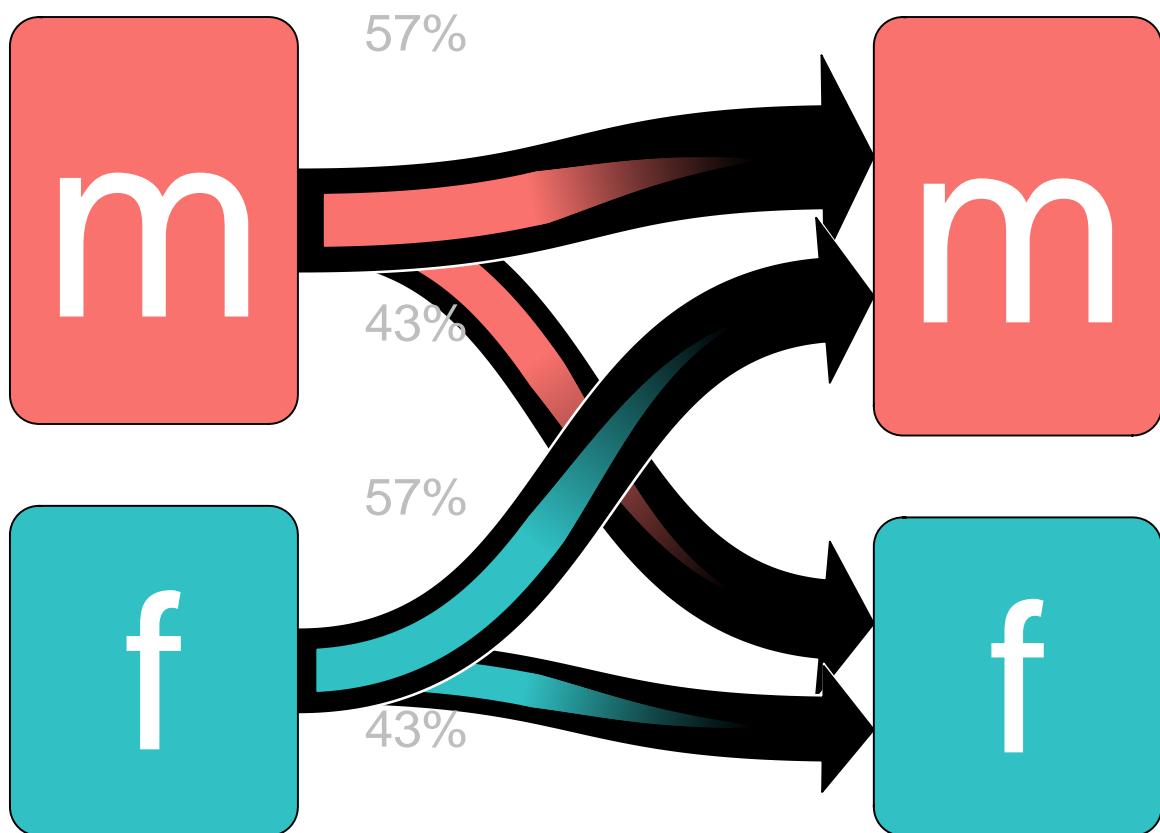
Next Speaker



Previous
Speaker

Next
Speaker

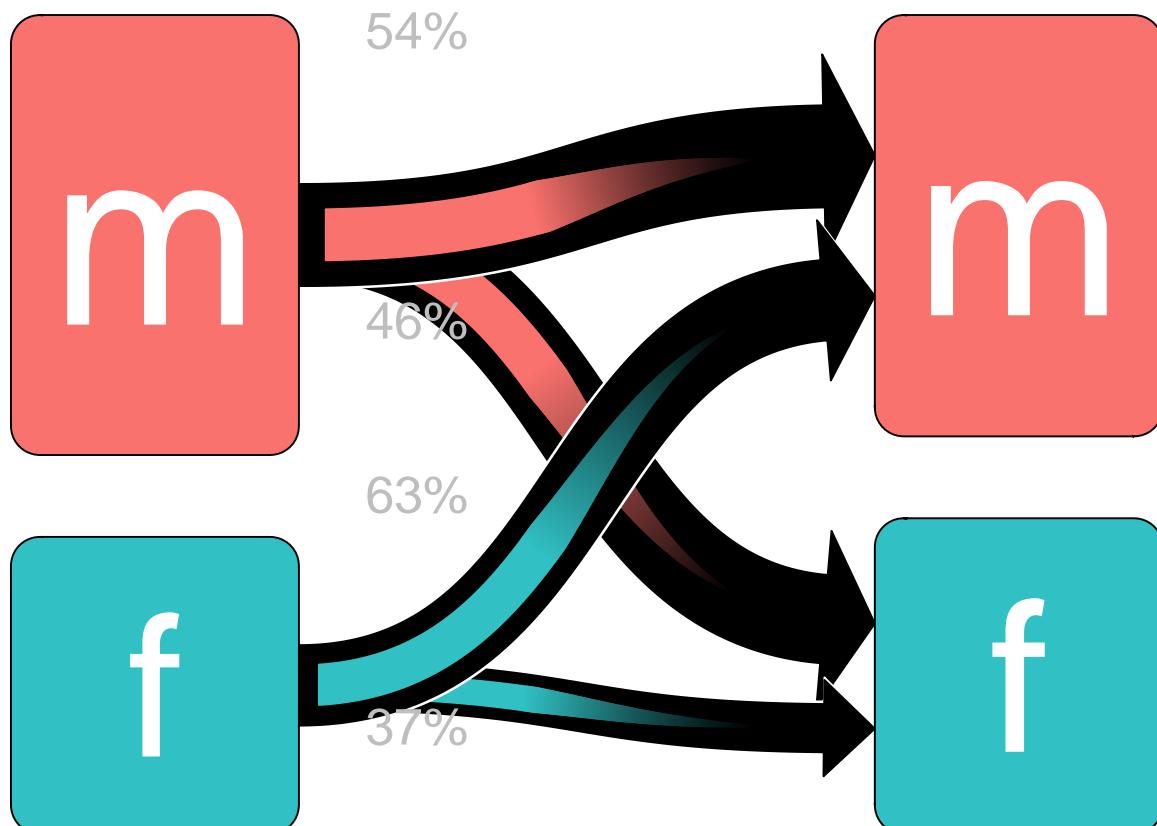
Chrono Trigger
Expected



Previous
Speaker

Dragon Age 2
Empirical

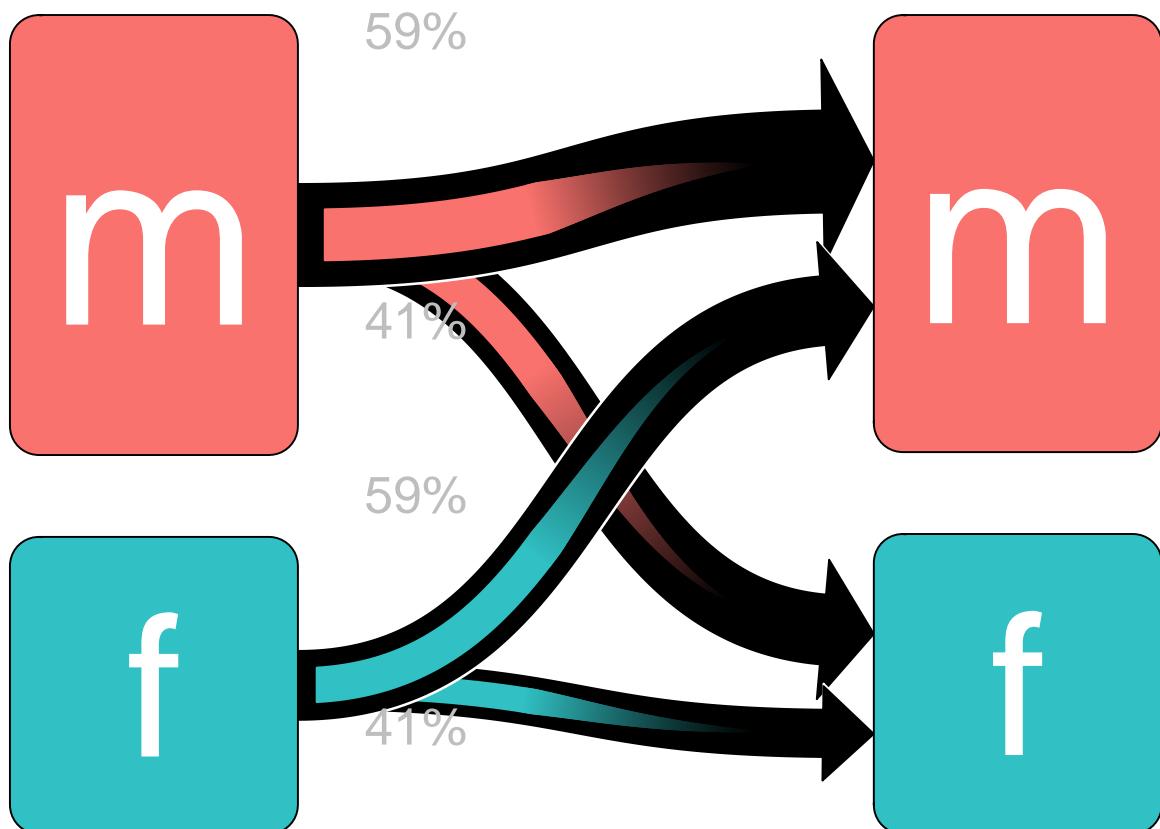
Next
Speaker



Previous
Speaker

Dragon Age 2
Expected

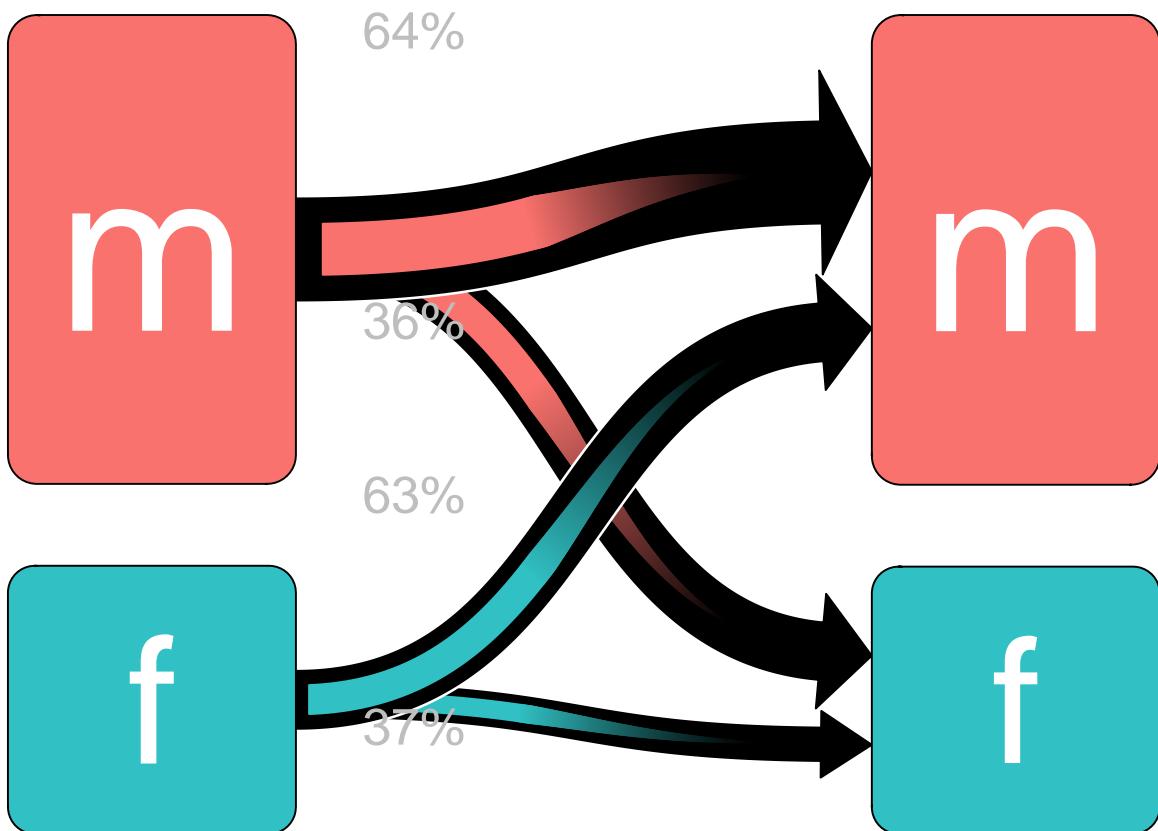
Next
Speaker



Previous
Speaker

Dragon Age Origins
Empirical

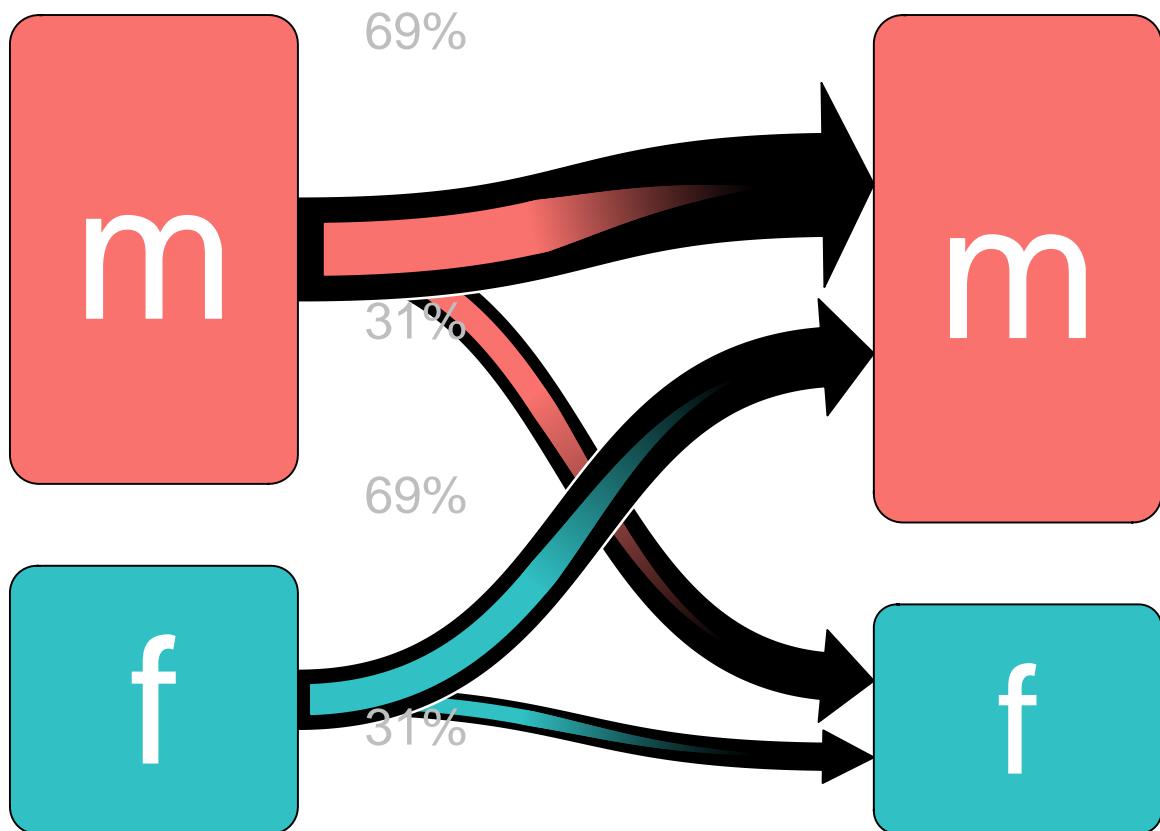
Next
Speaker



Previous
Speaker

Dragon Age Origins
Expected

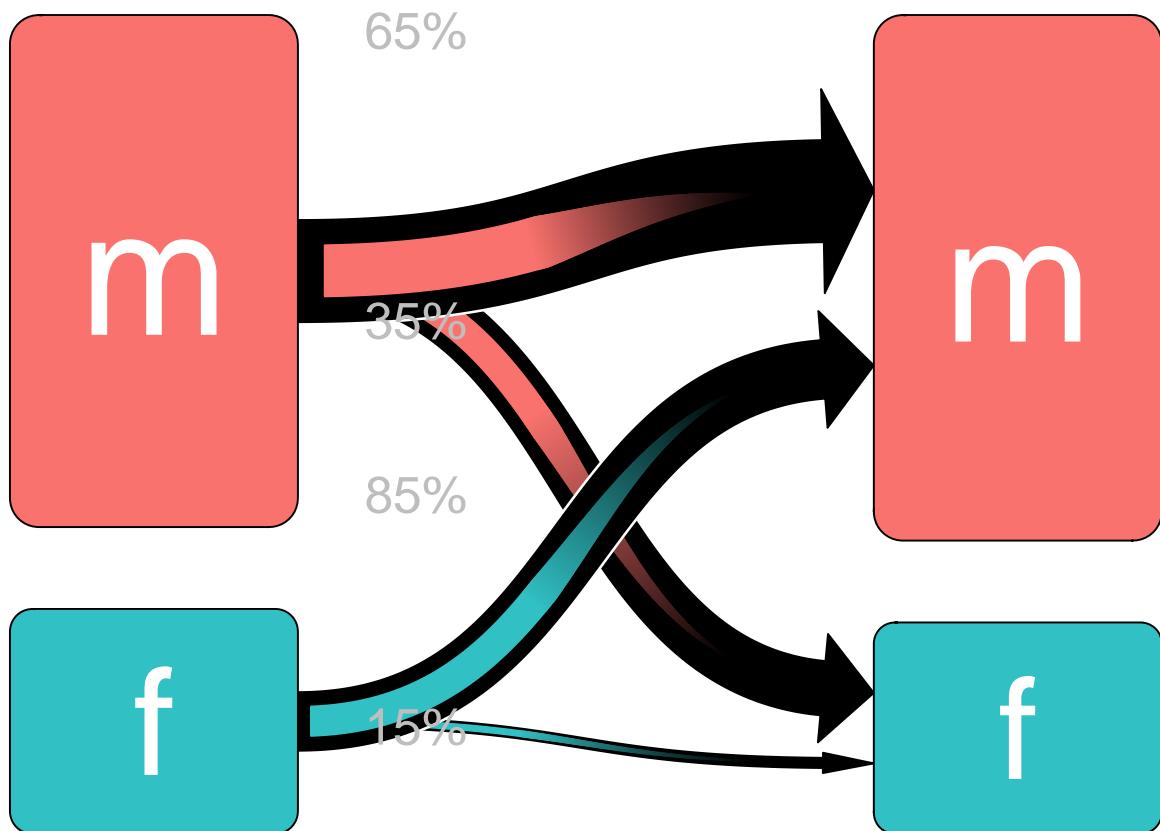
Next
Speaker



Previous Speaker

The Elder Scrolls V Skyrim
Empirical

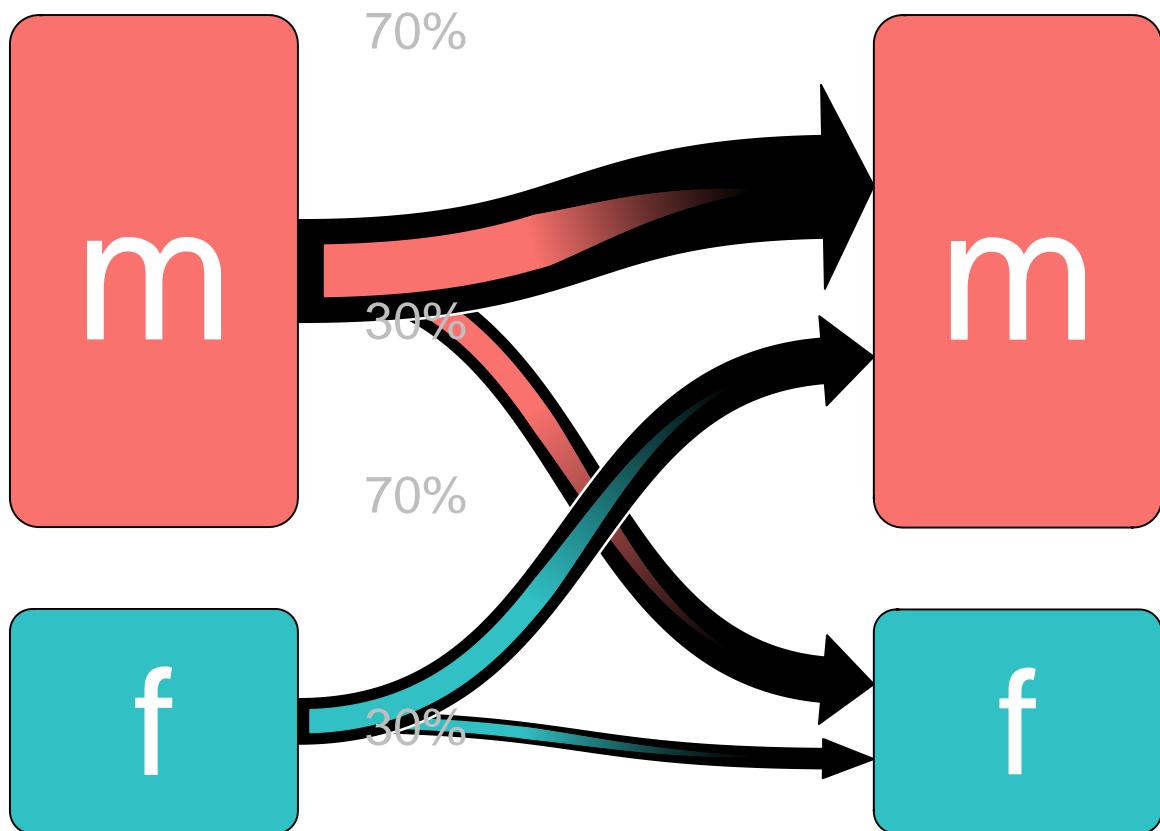
Next Speaker



Previous Speaker

The Elder Scrolls V Skyrim
Expected

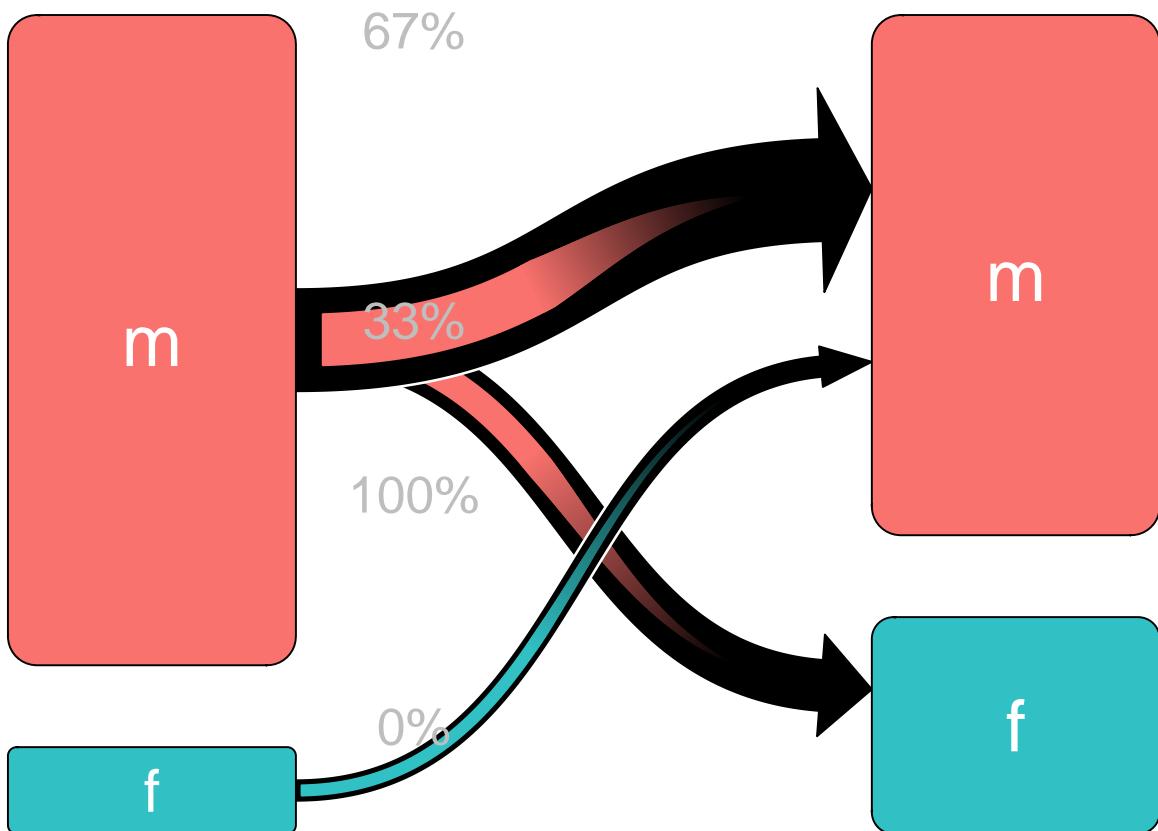
Next Speaker



Previous Speaker

Final Fantasy
Empirical

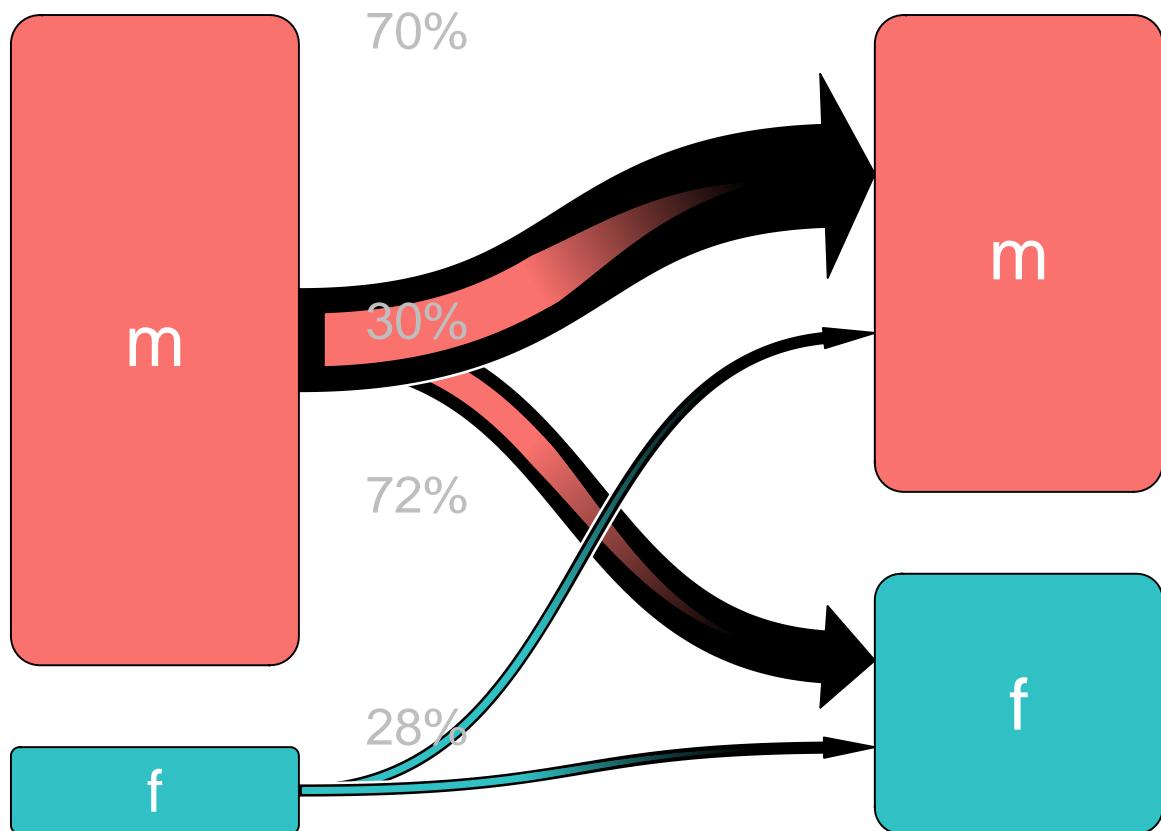
Next Speaker



Previous Speaker

Final Fantasy
Expected

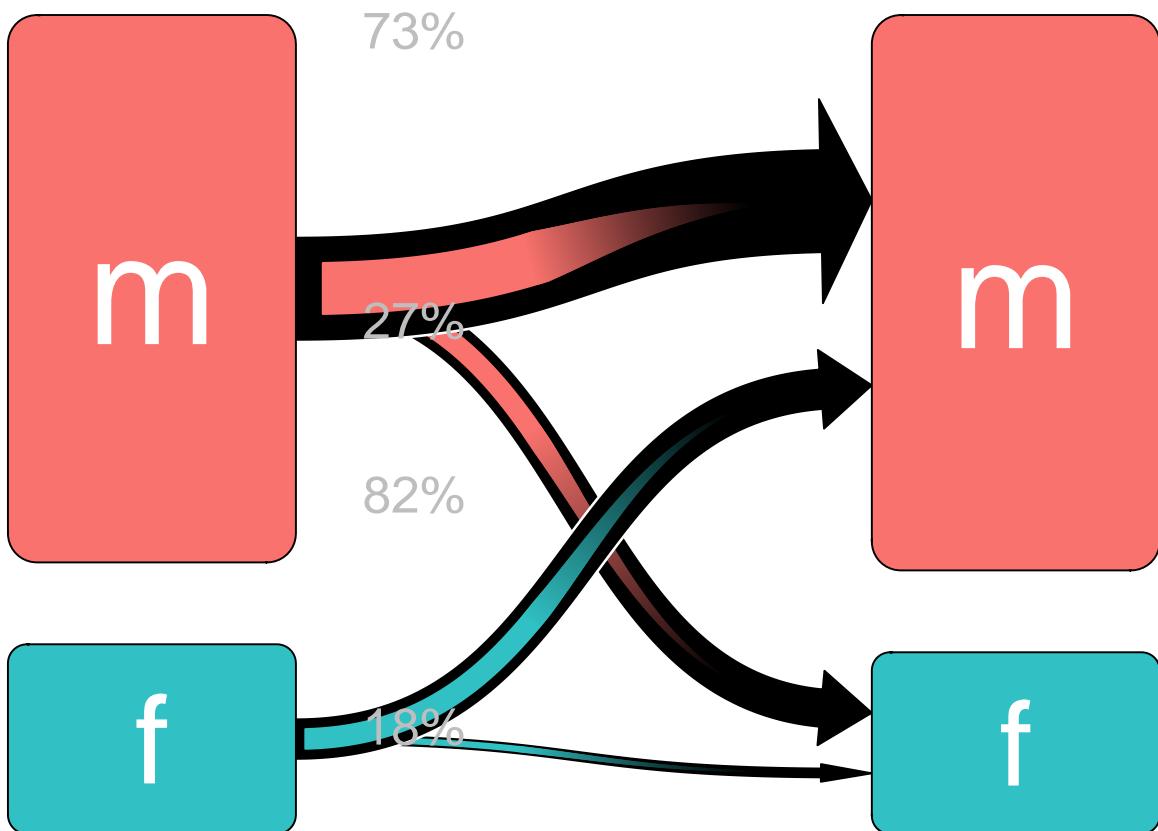
Next Speaker



Previous
Speaker

Final Fantasy II
Empirical

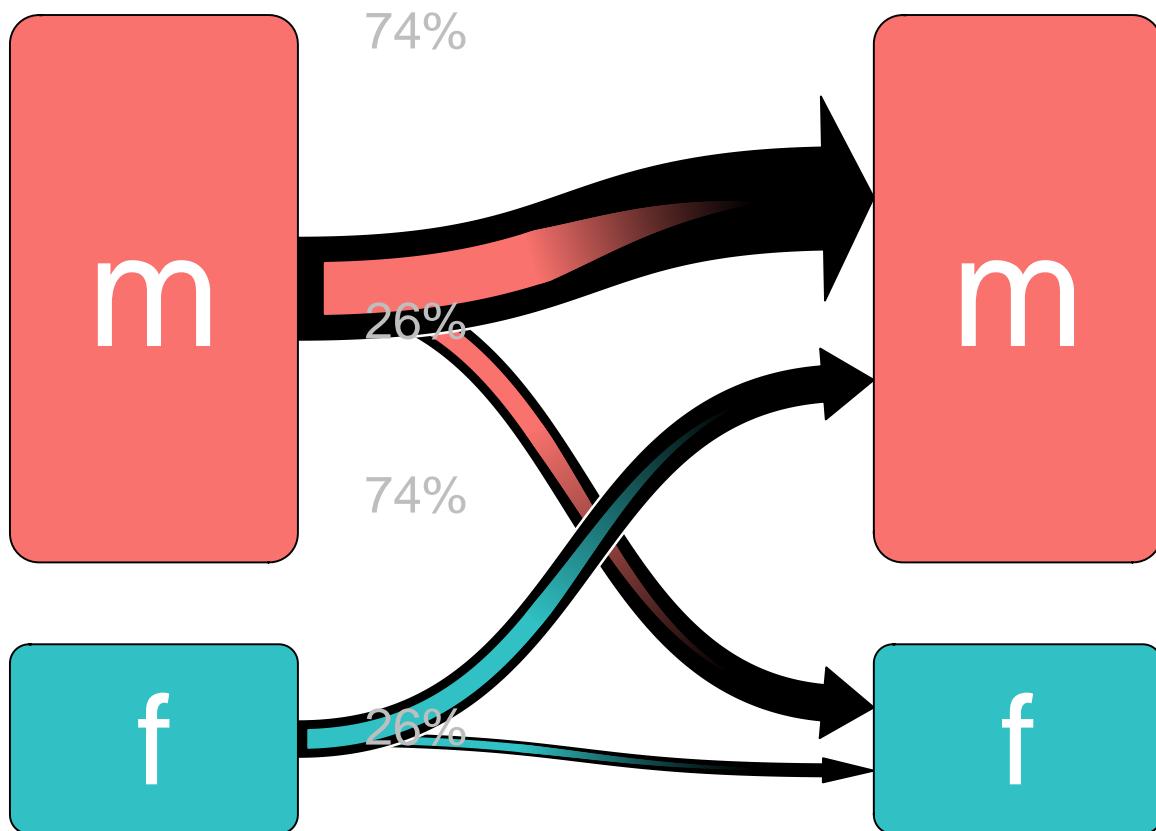
Next
Speaker



Previous
Speaker

Final Fantasy II
Expected

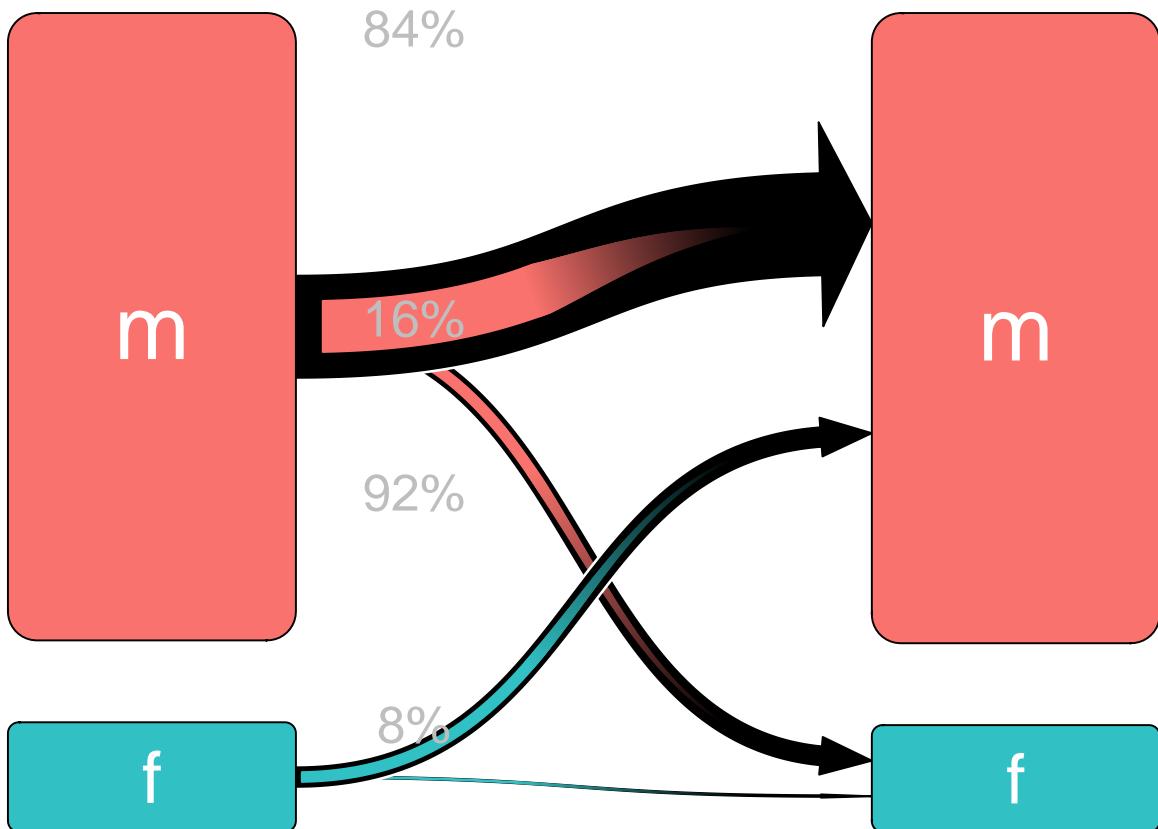
Next
Speaker



Previous Speaker

Final Fantasy IV
Empirical

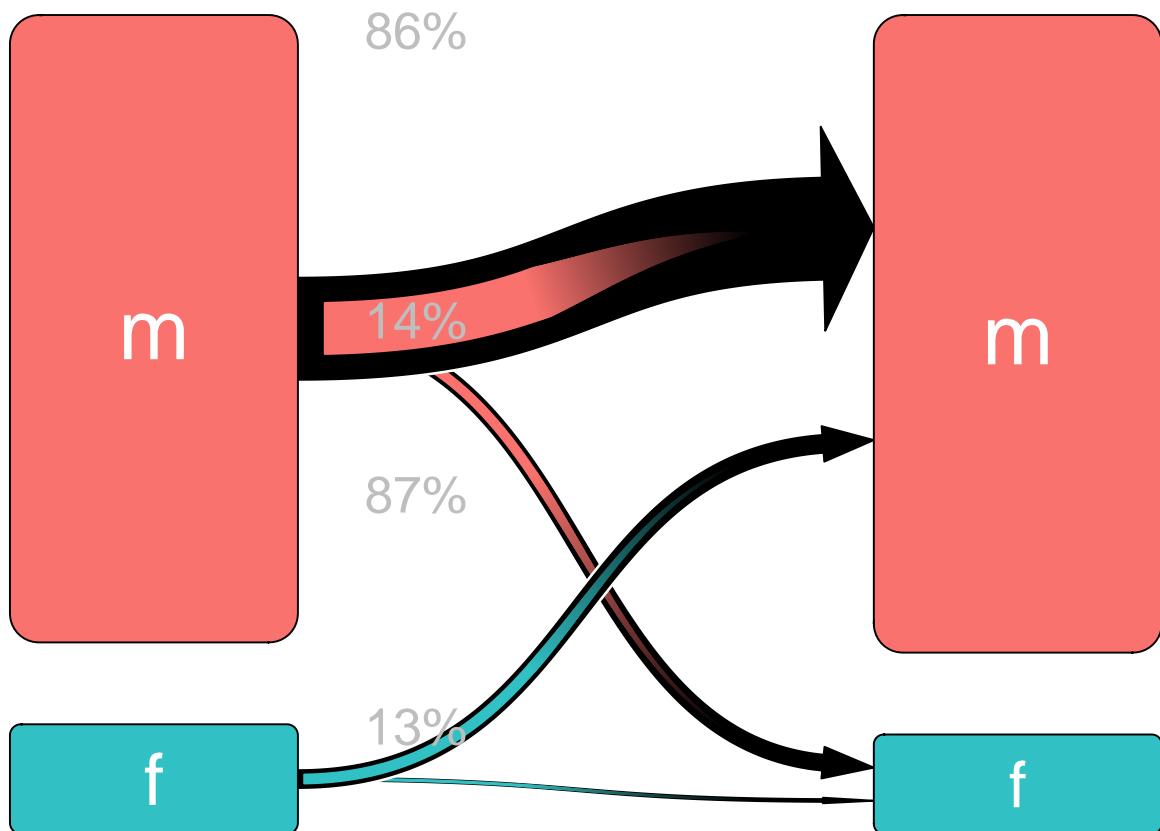
Next Speaker



Previous
Speaker

Final Fantasy IV
Expected

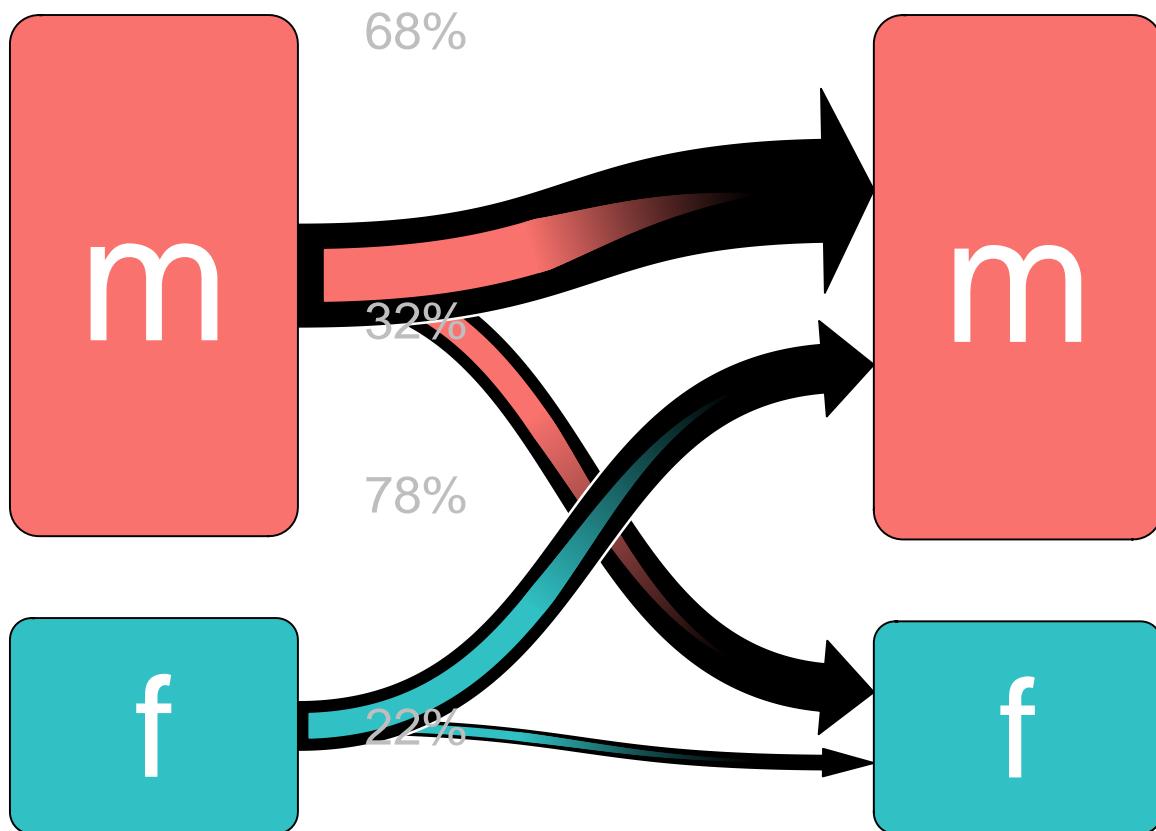
Next
Speaker



Previous
Speaker

Final Fantasy IX
Empirical

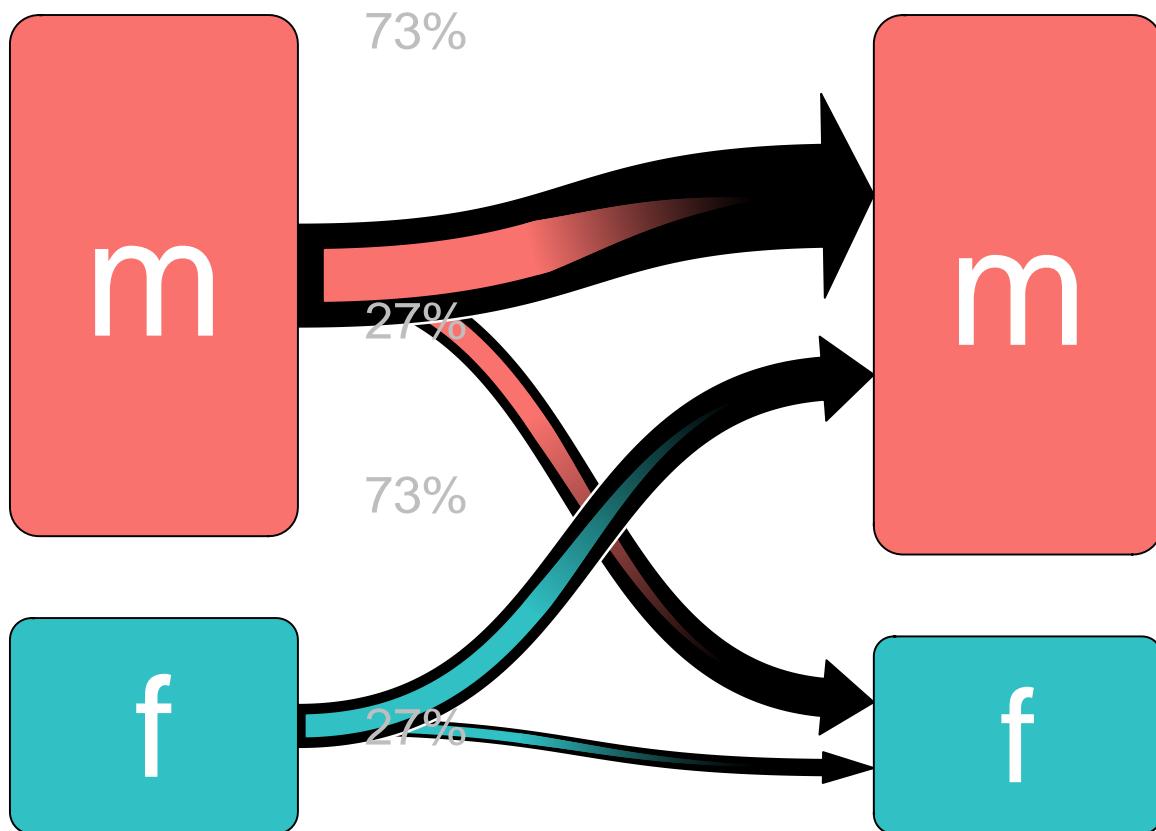
Next
Speaker



Previous
Speaker

Final Fantasy IX
Expected

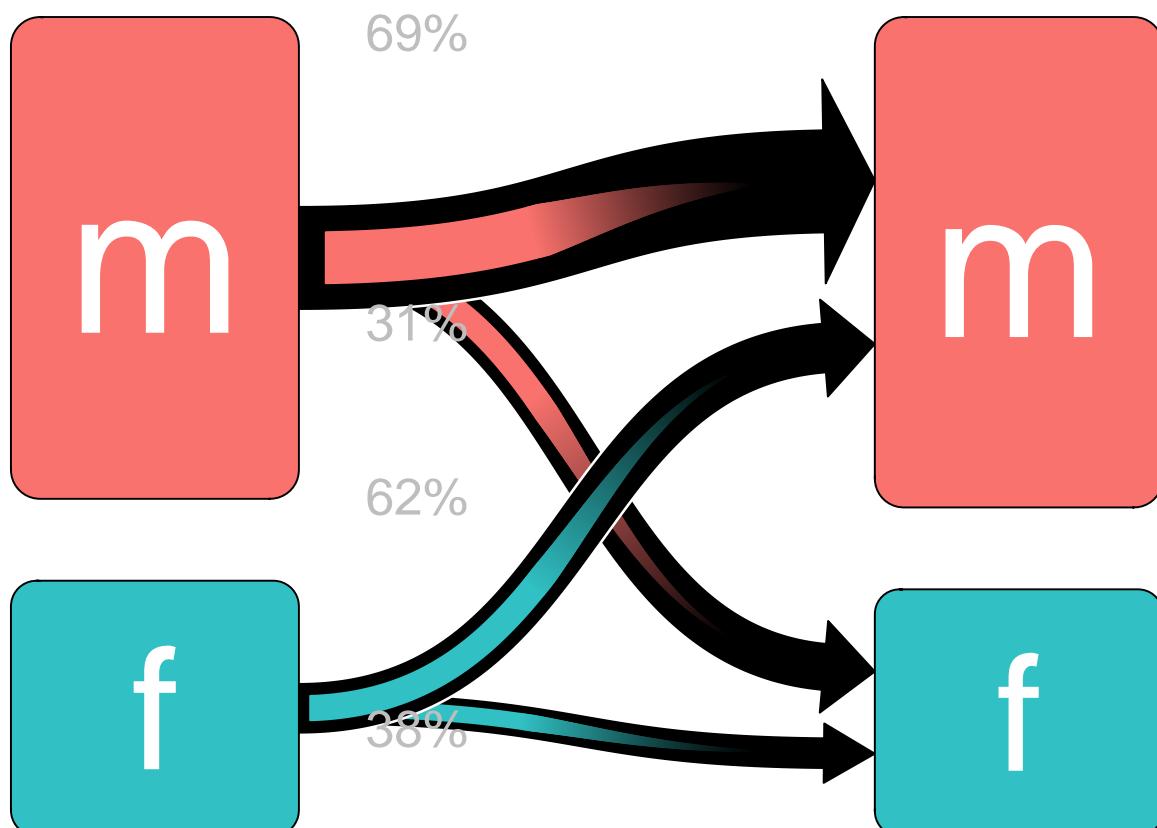
Next
Speaker



Previous
Speaker

Next
Speaker

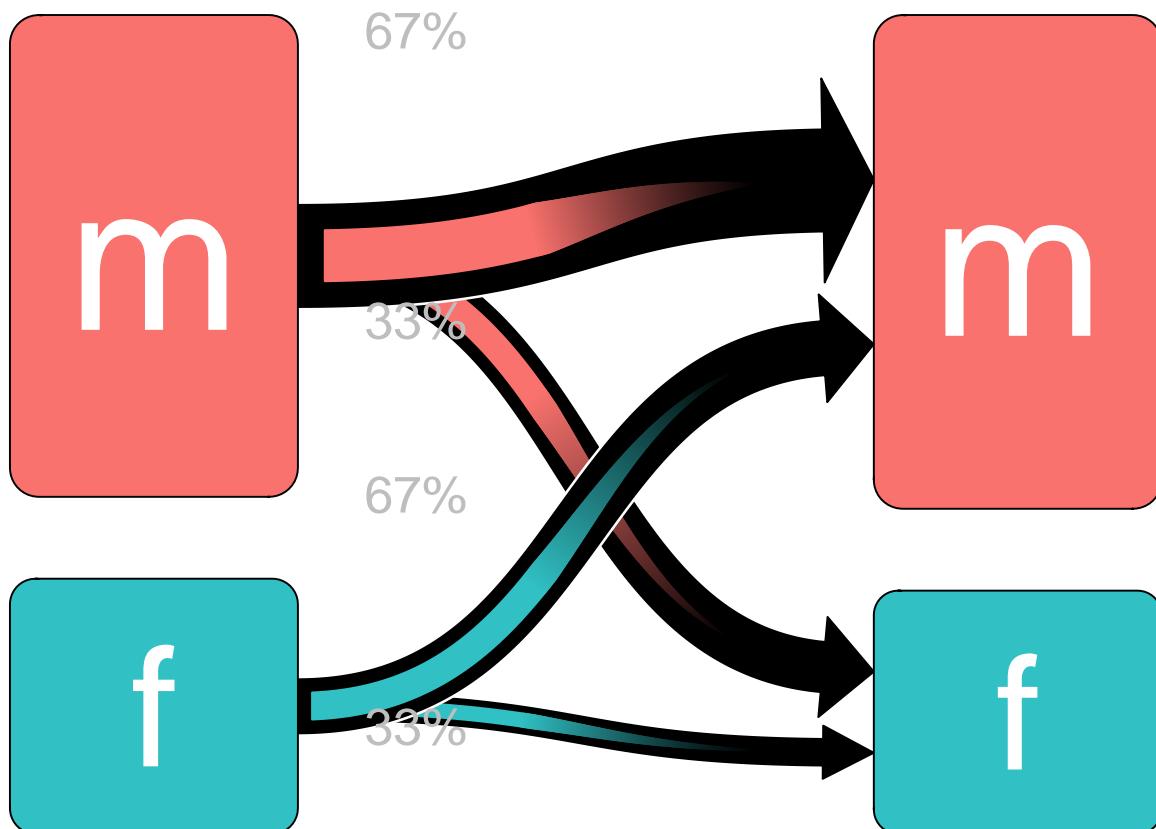
Final Fantasy V
Empirical



Previous
Speaker

Final Fantasy V
Expected

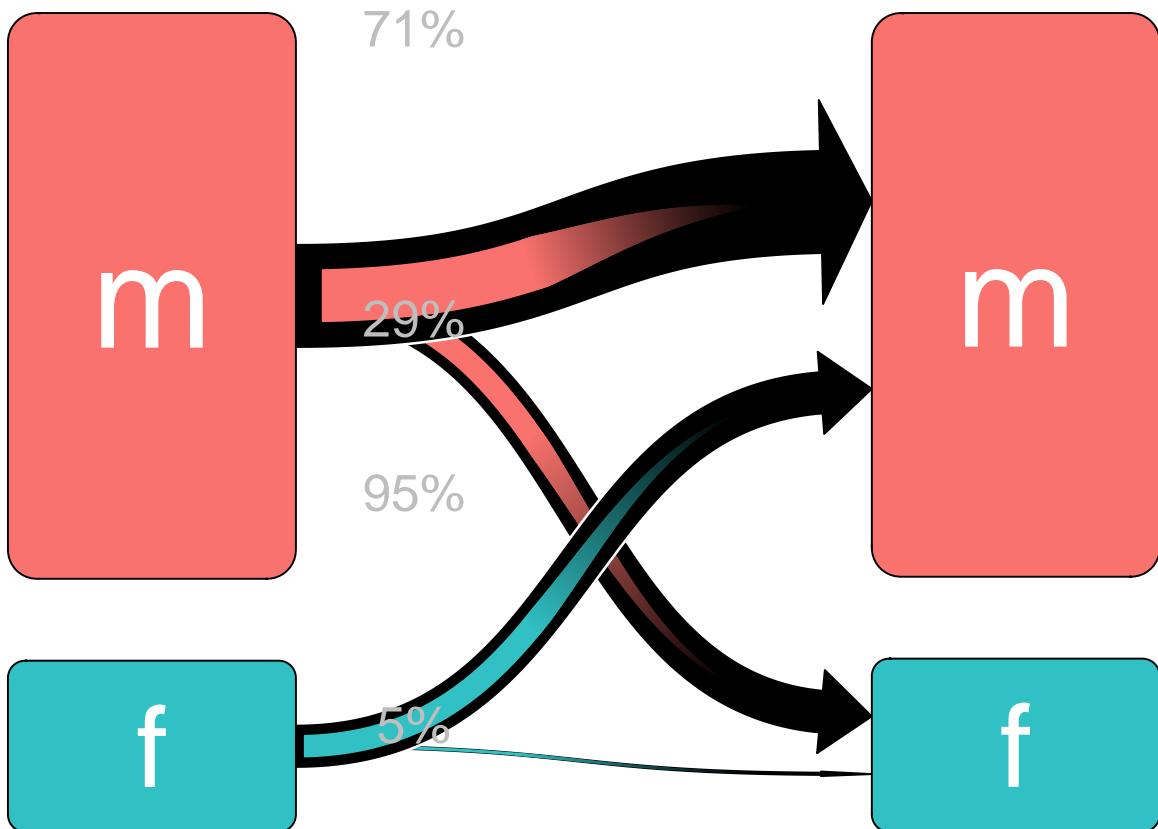
Next
Speaker



Previous
Speaker

Final Fantasy VI
Empirical

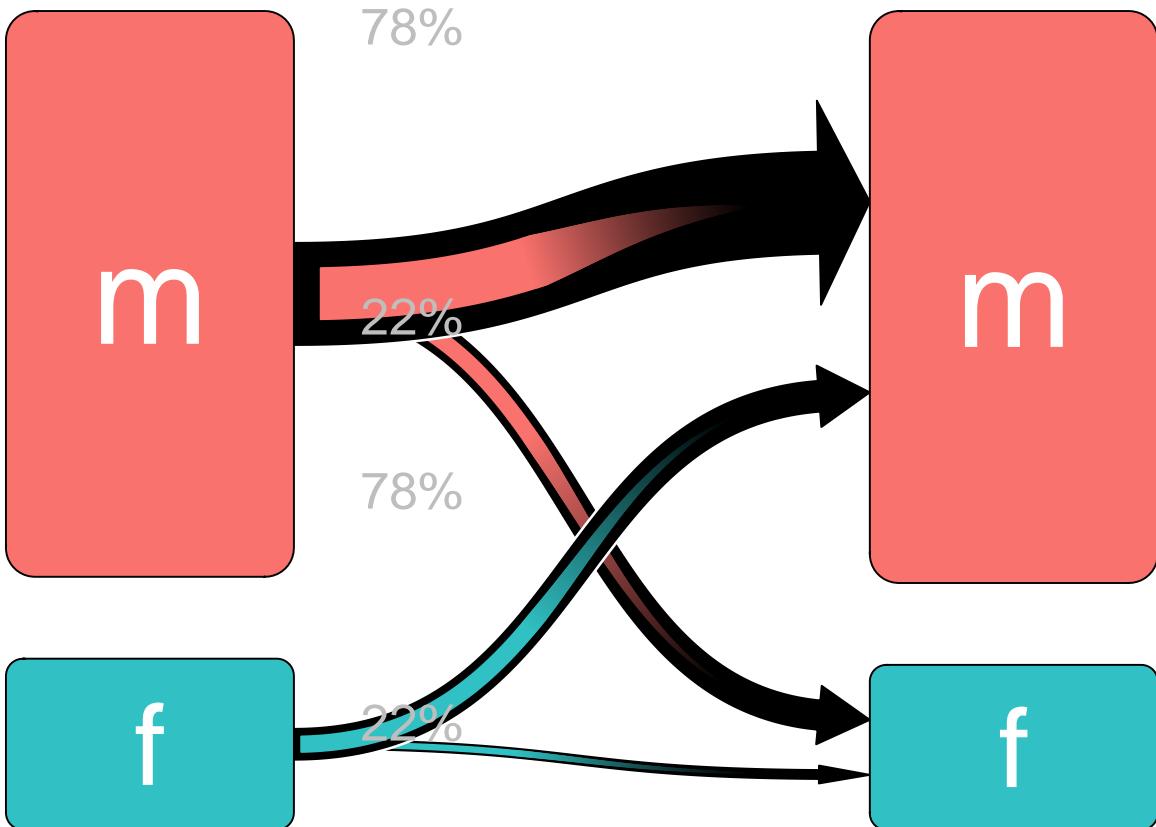
Next
Speaker



Previous
Speaker

Final Fantasy VI
Expected

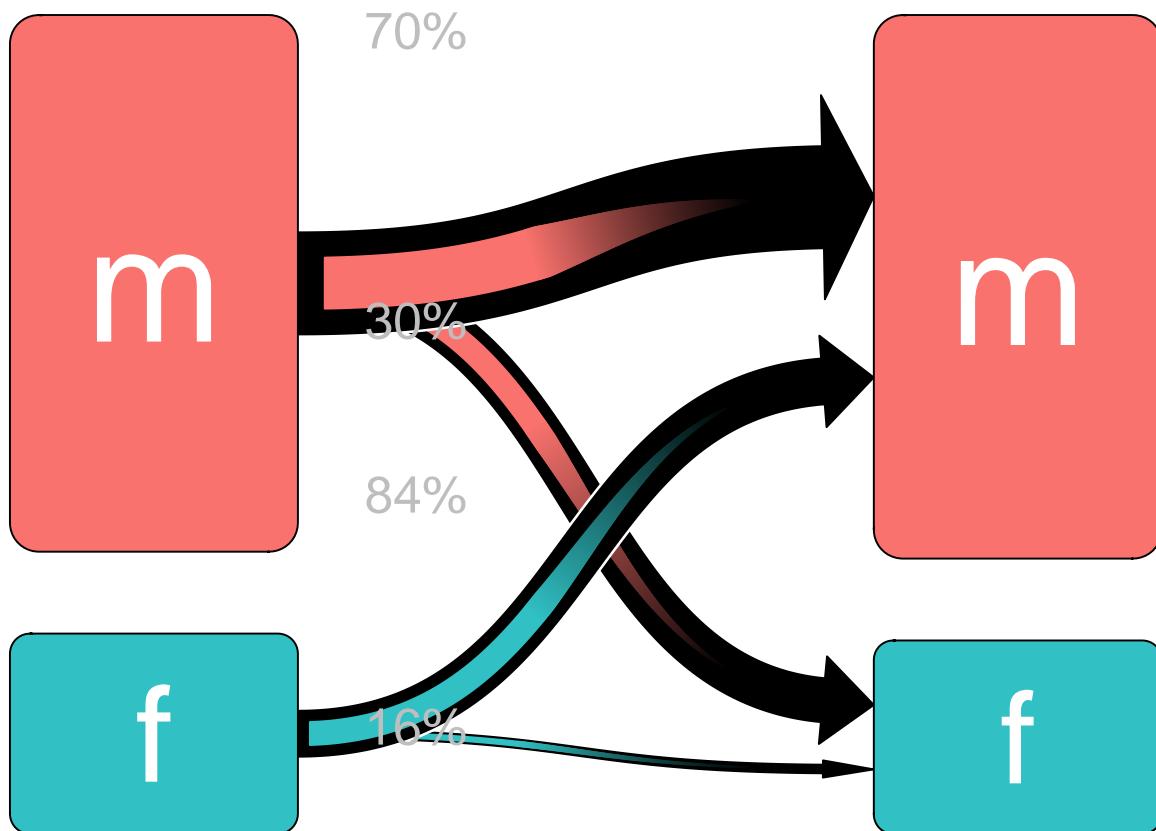
Next
Speaker



Previous
Speaker

Final Fantasy VII
Empirical

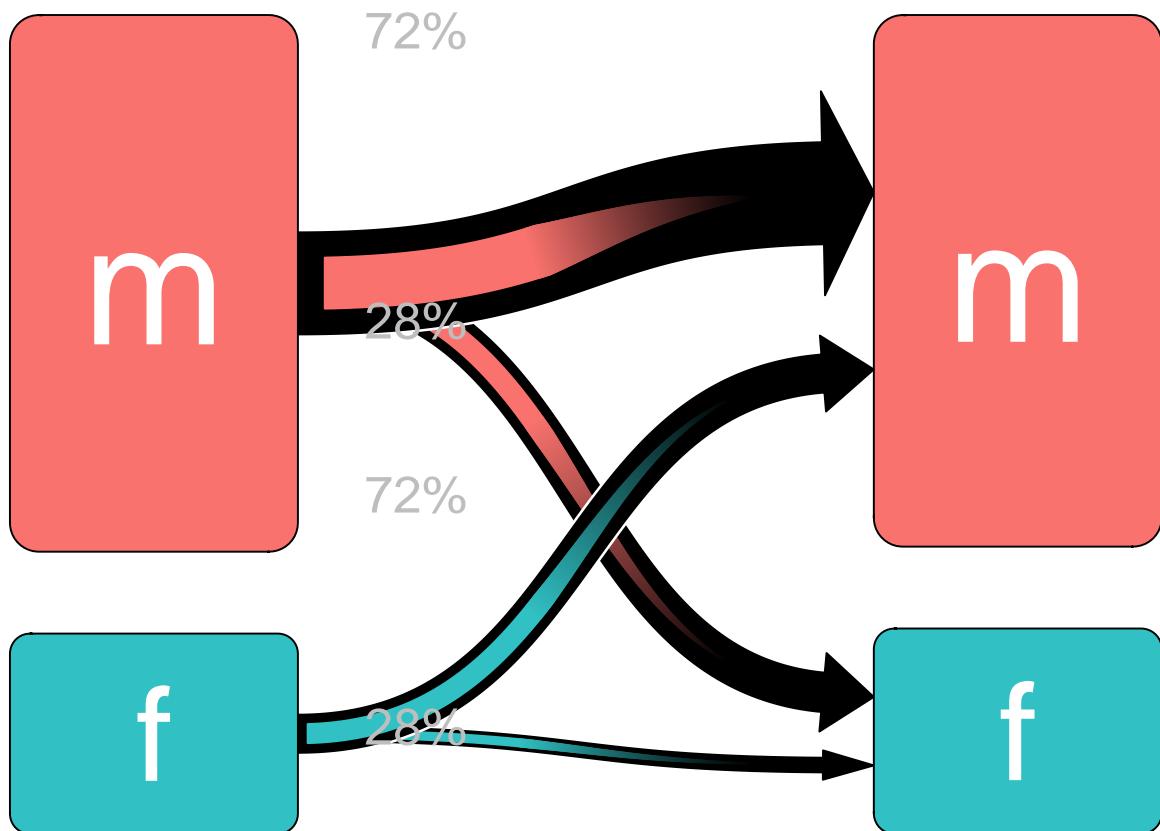
Next
Speaker



Previous
Speaker

Final Fantasy VII
Expected

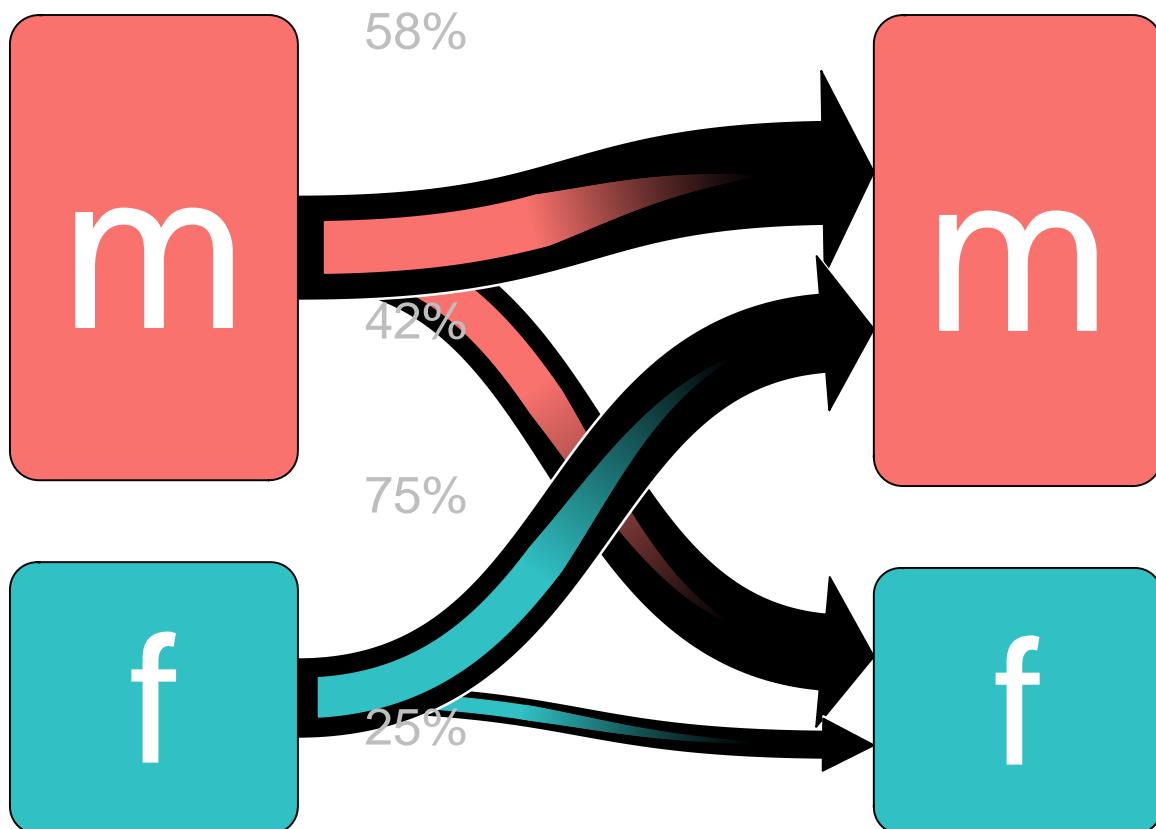
Next
Speaker



Previous
Speaker

Final Fantasy VII Remake
Empirical

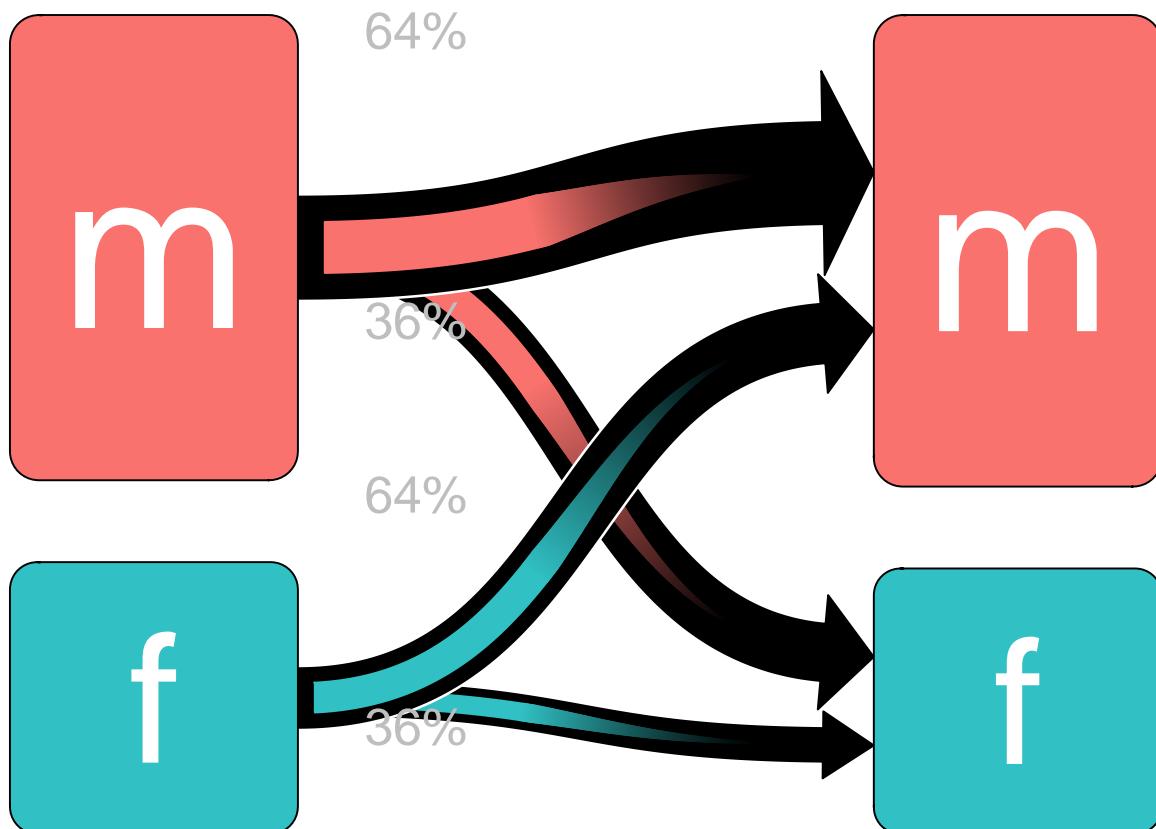
Next
Speaker



Previous
Speaker

Final Fantasy VII Remake
Expected

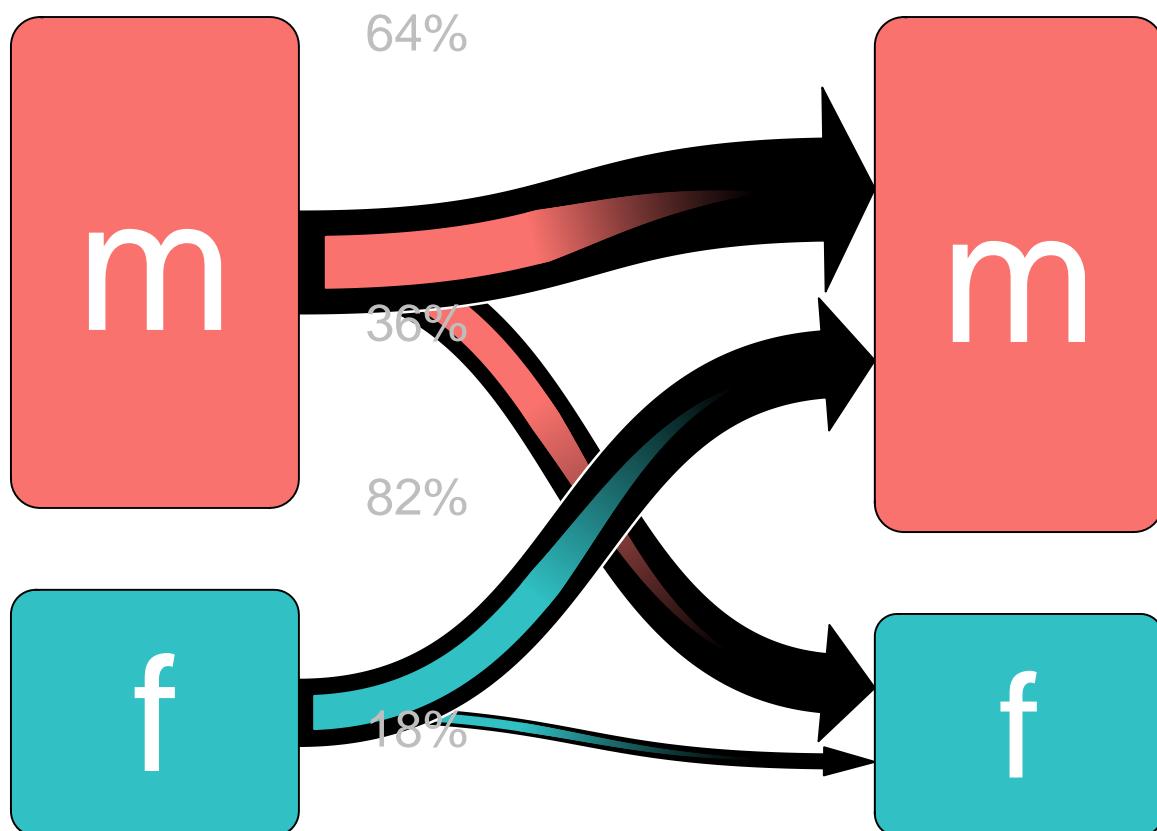
Next
Speaker



Previous
Speaker

Next
Speaker

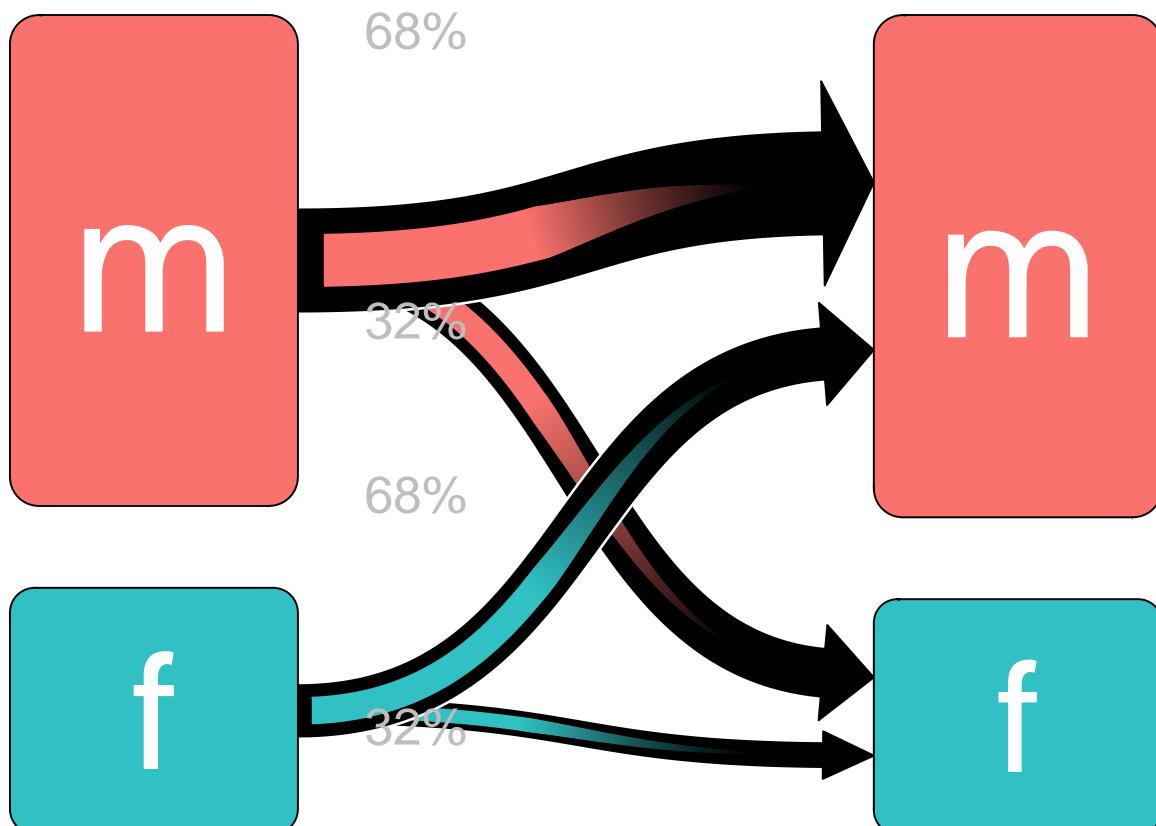
Final Fantasy VIII
Empirical



Previous
Speaker

Final Fantasy VIII
Expected

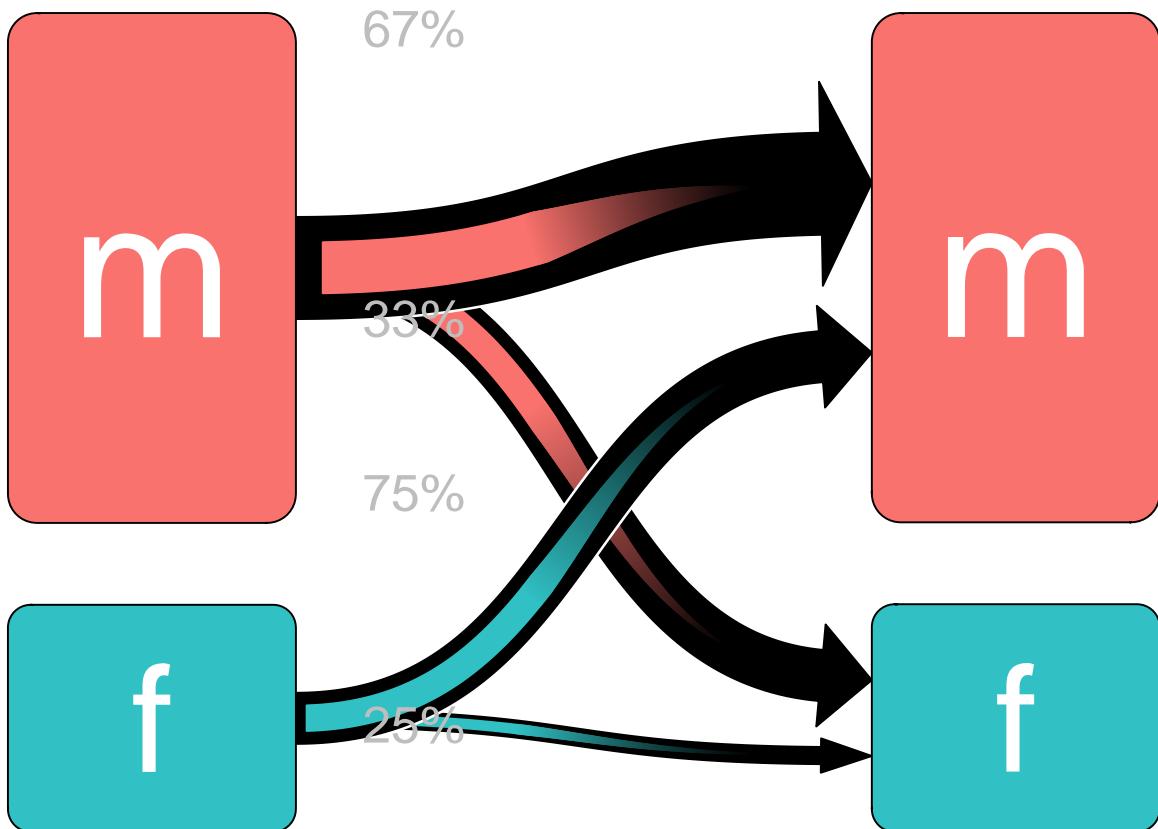
Next
Speaker



Previous
Speaker

Final Fantasy X
Empirical

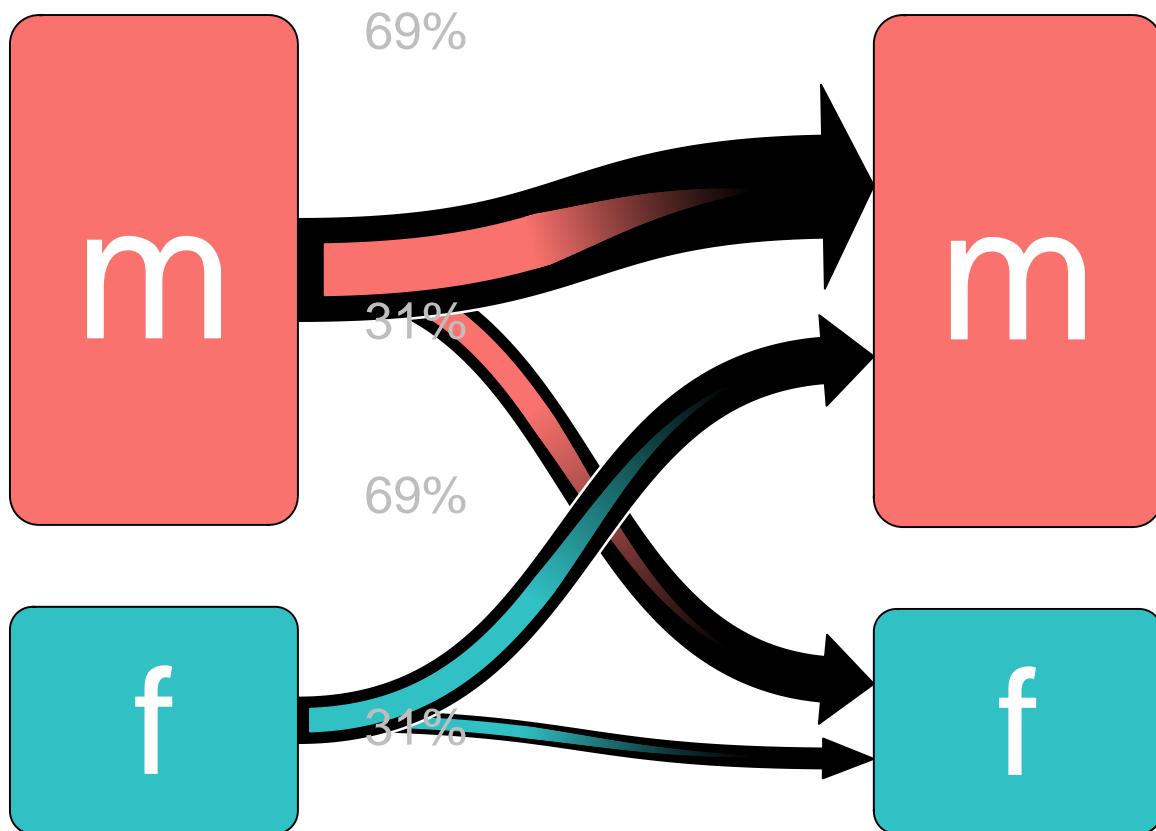
Next
Speaker



Previous
Speaker

Final Fantasy X
Expected

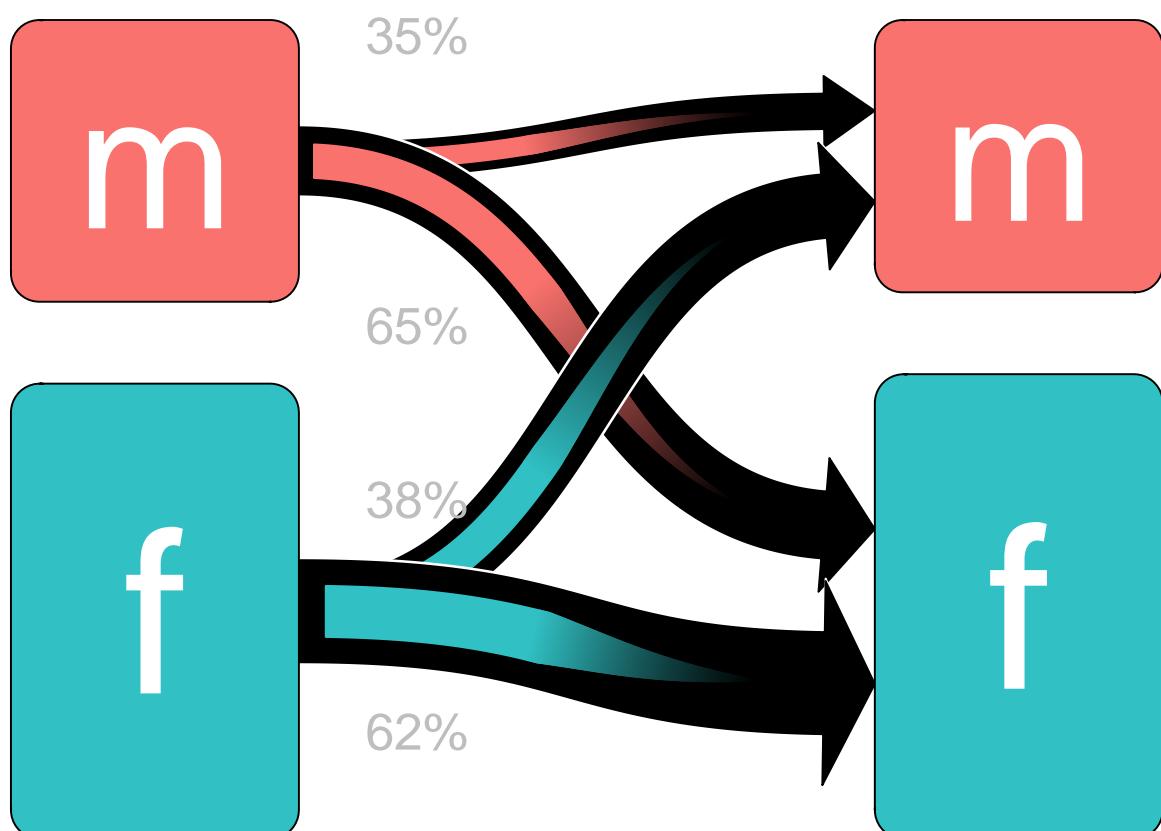
Next
Speaker



Previous
Speaker

Next
Speaker

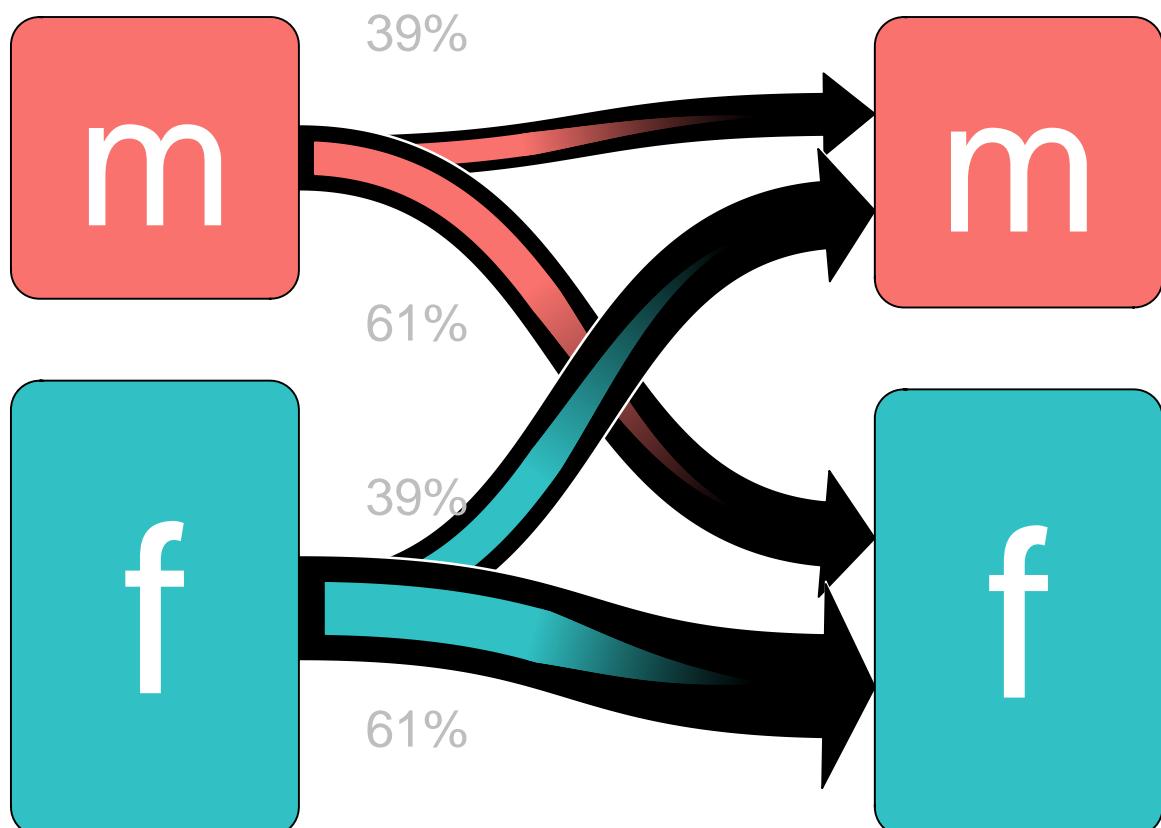
Final Fantasy X-2
Empirical



Previous
Speaker

Next
Speaker

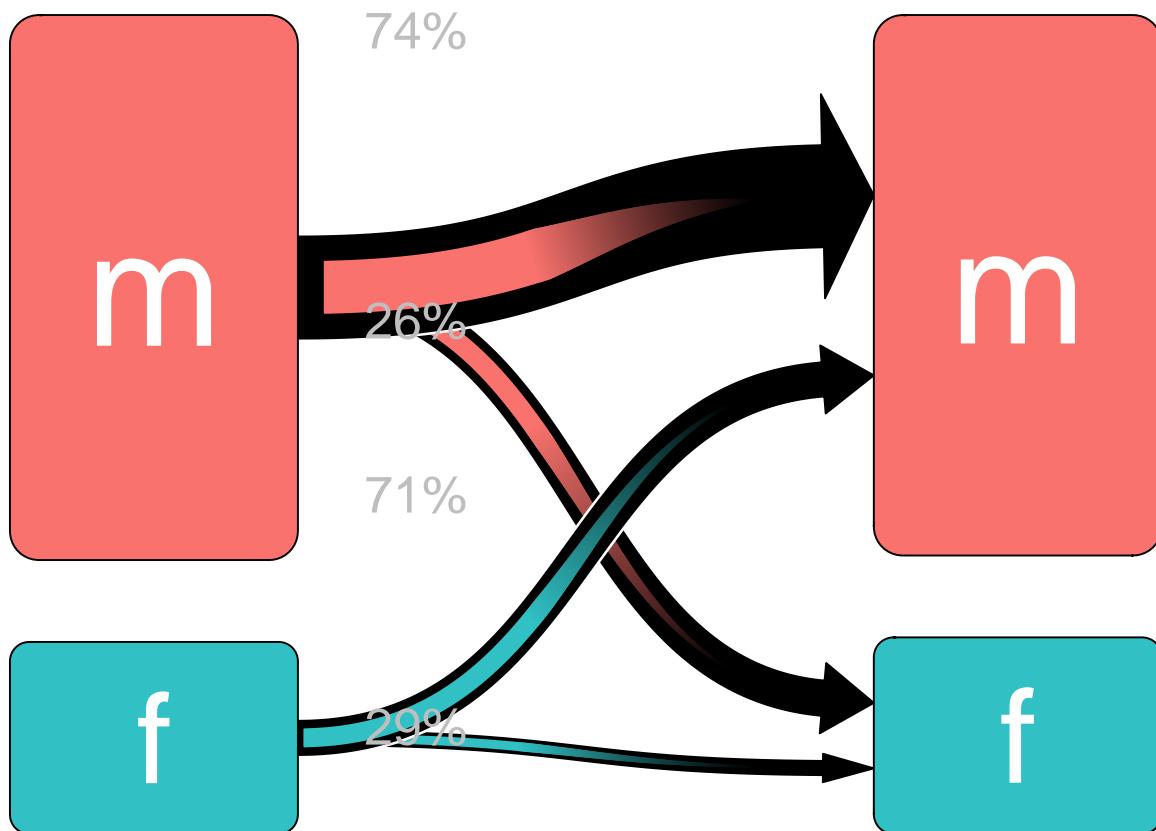
Final Fantasy X-2
Expected



Previous
Speaker

Final Fantasy XII
Empirical

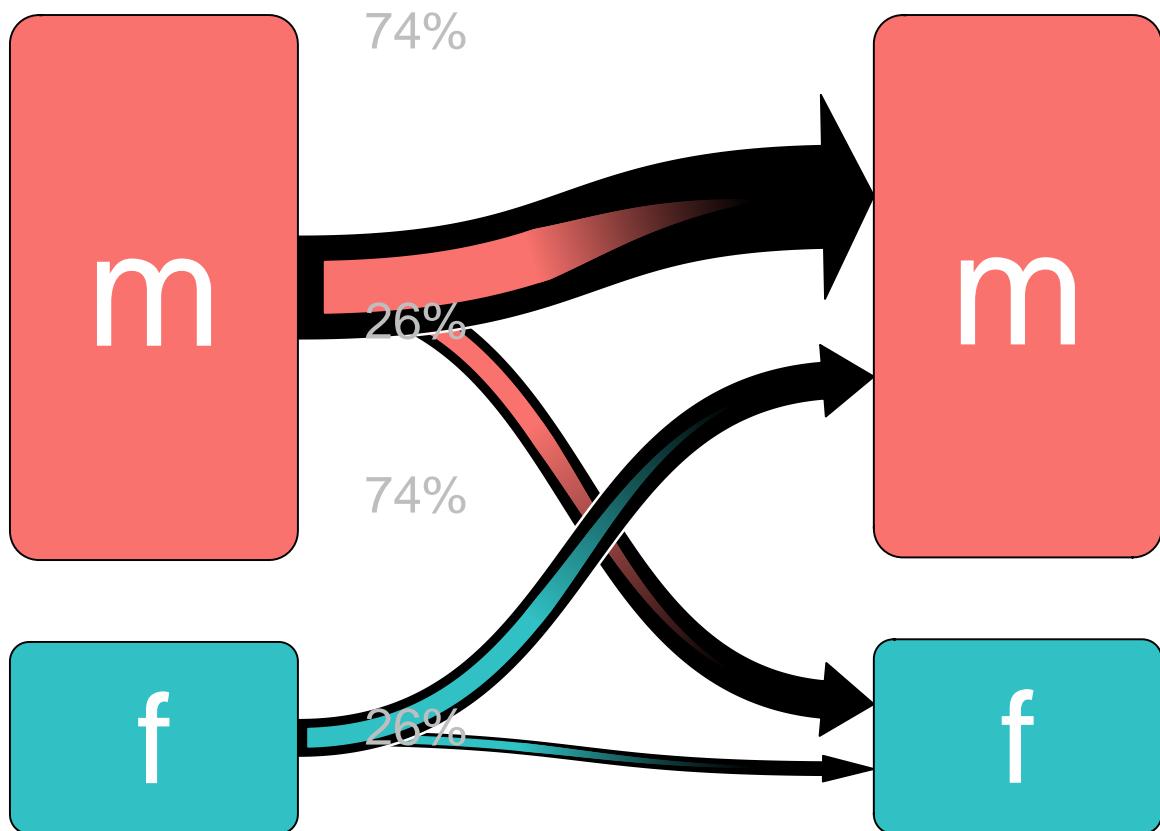
Next
Speaker



Previous
Speaker

Final Fantasy XII
Expected

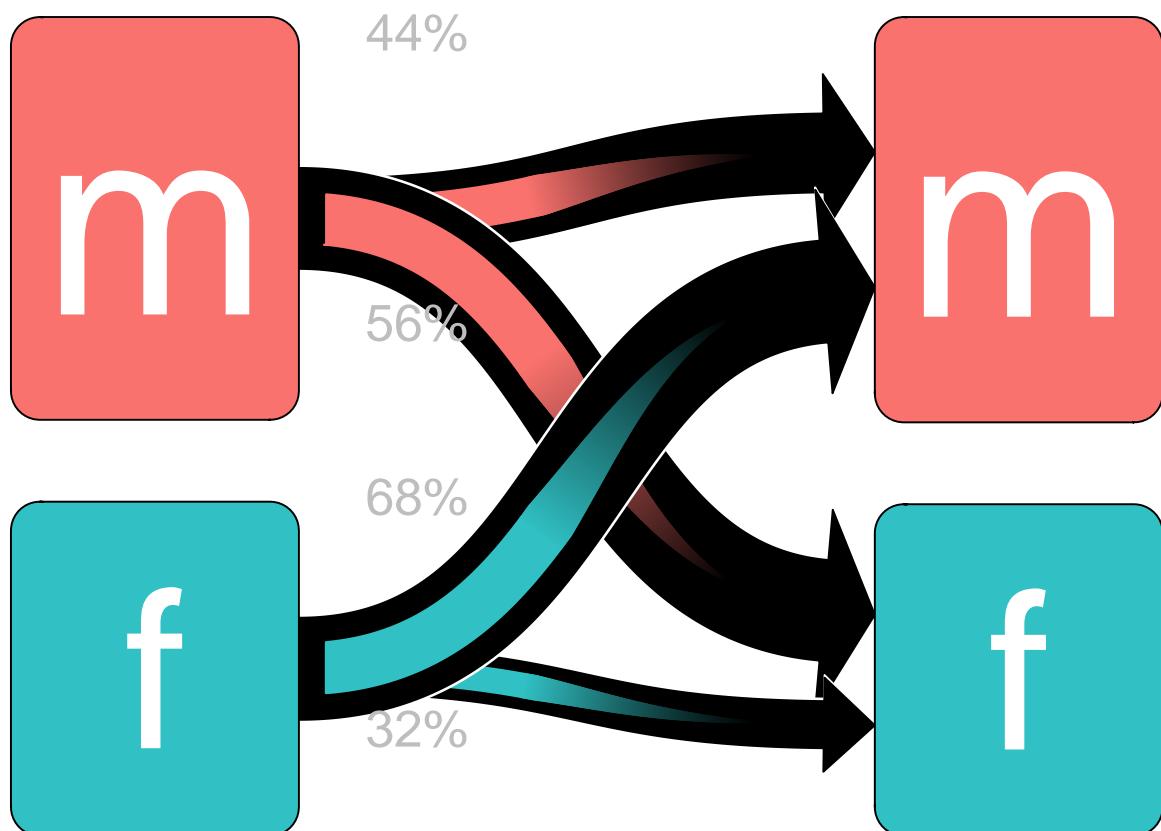
Next
Speaker



Previous
Speaker

Next
Speaker

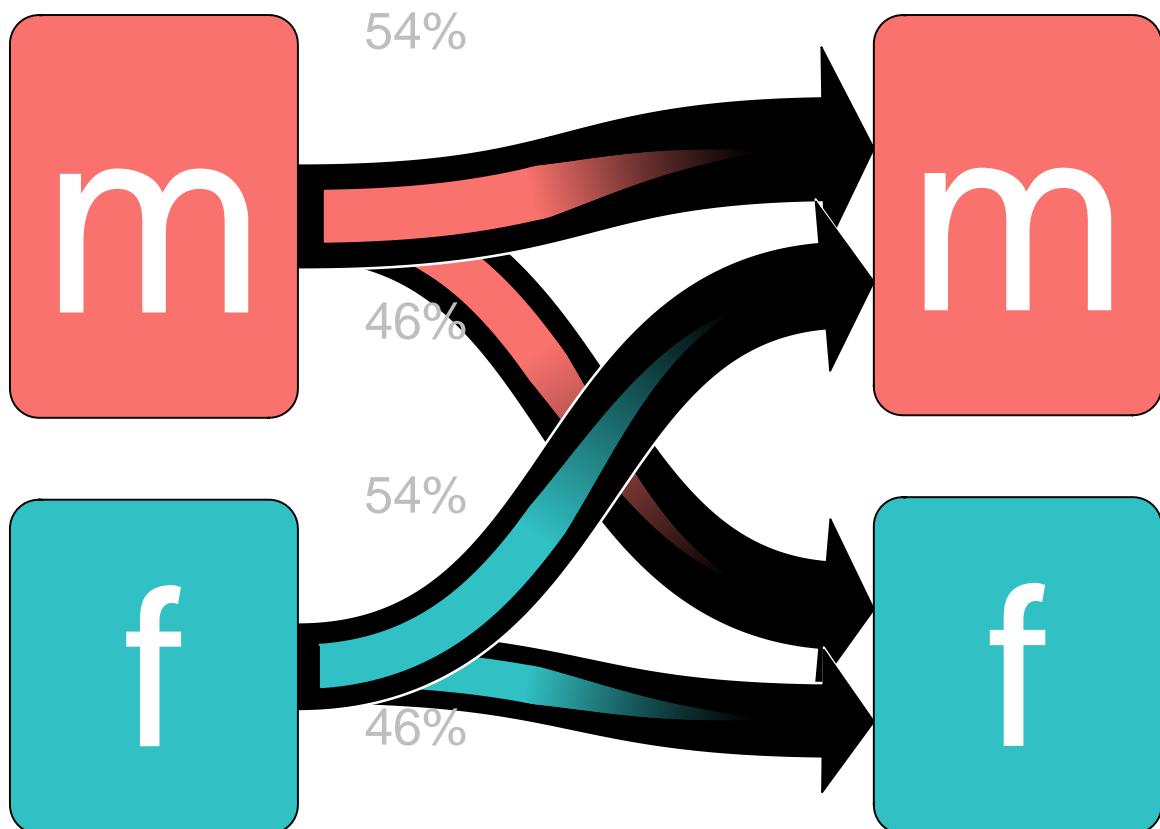
Final Fantasy XIII
Empirical



Previous
Speaker

Final Fantasy XIII
Expected

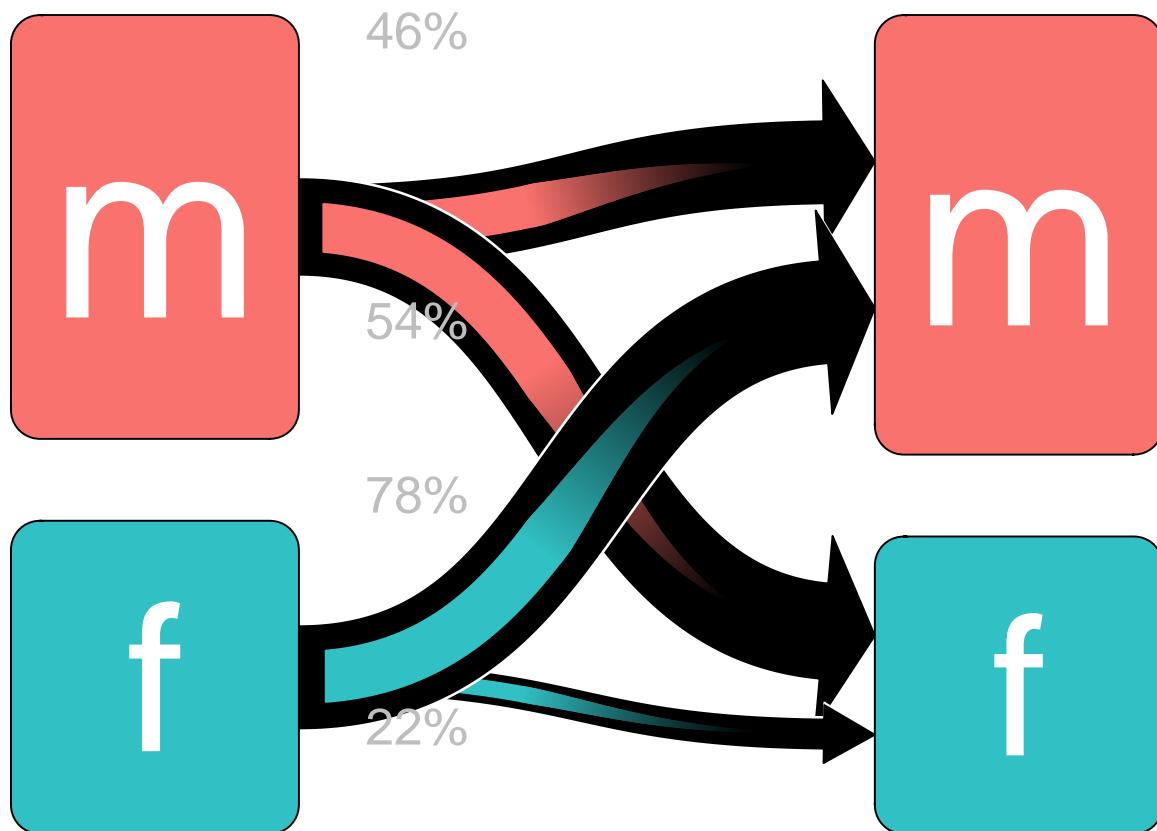
Next
Speaker



Previous
Speaker

Final Fantasy XIII–2
Empirical

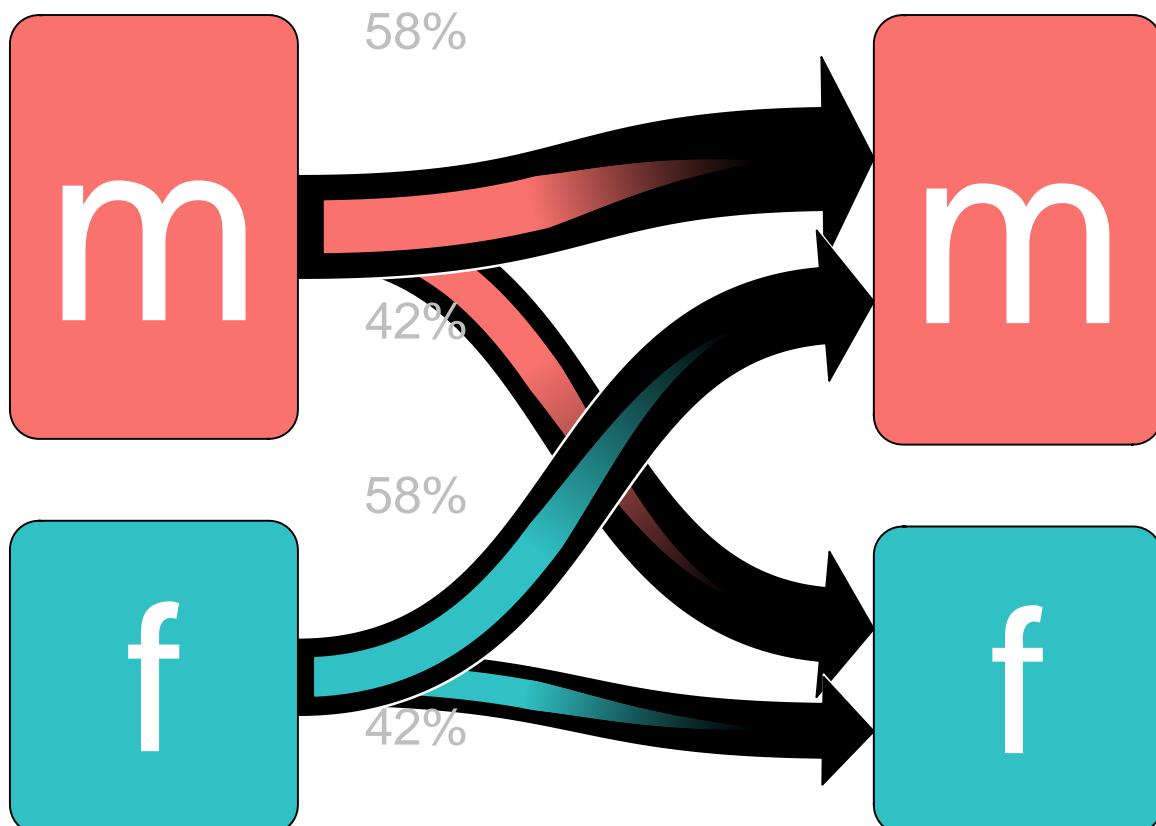
Next
Speaker



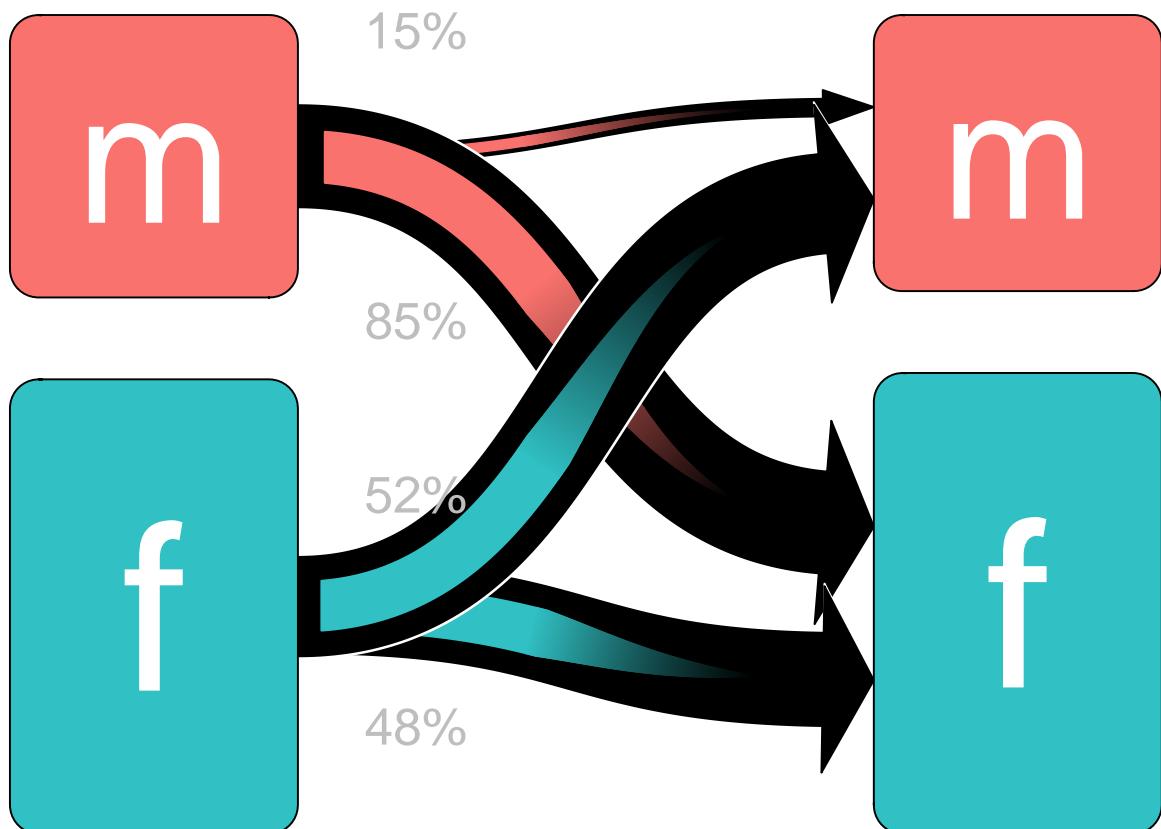
Previous
Speaker

Final Fantasy XIII-2
Expected

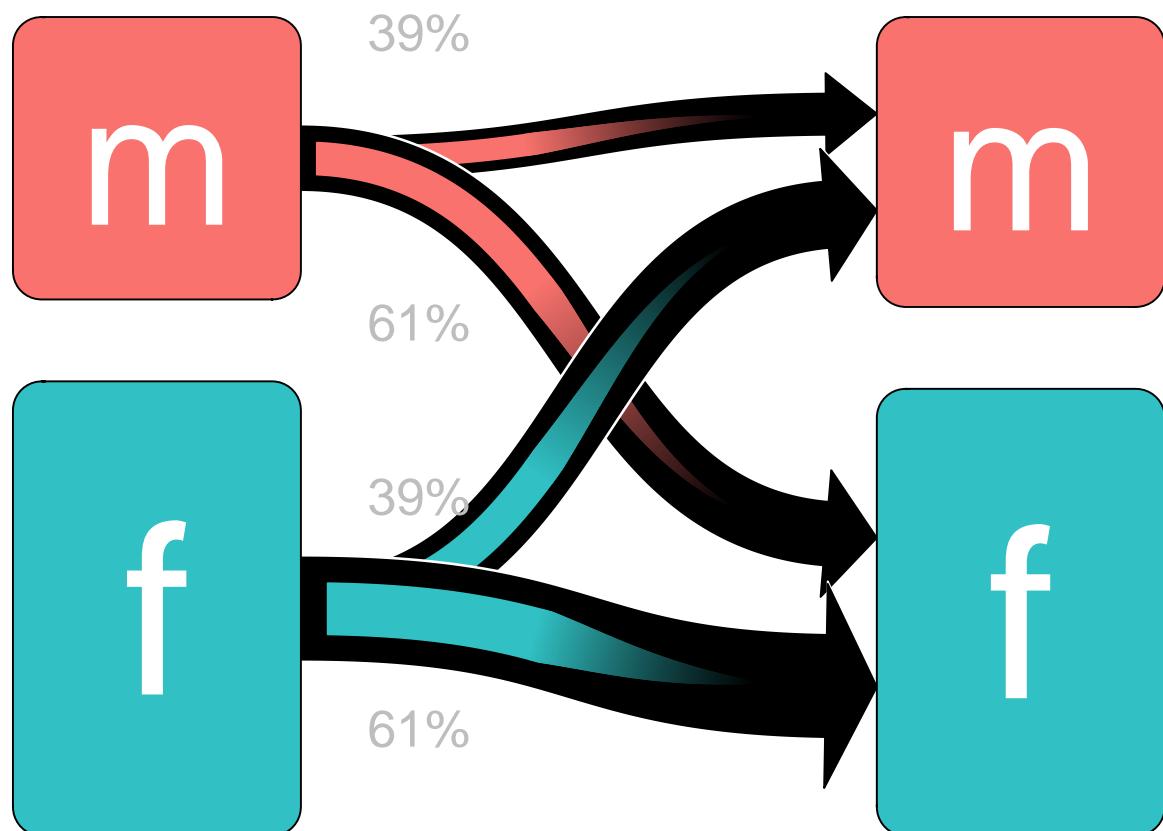
Next
Speaker



Previous Speaker Lightning Returns Final Fantasy XIII Next Speaker
Empirical



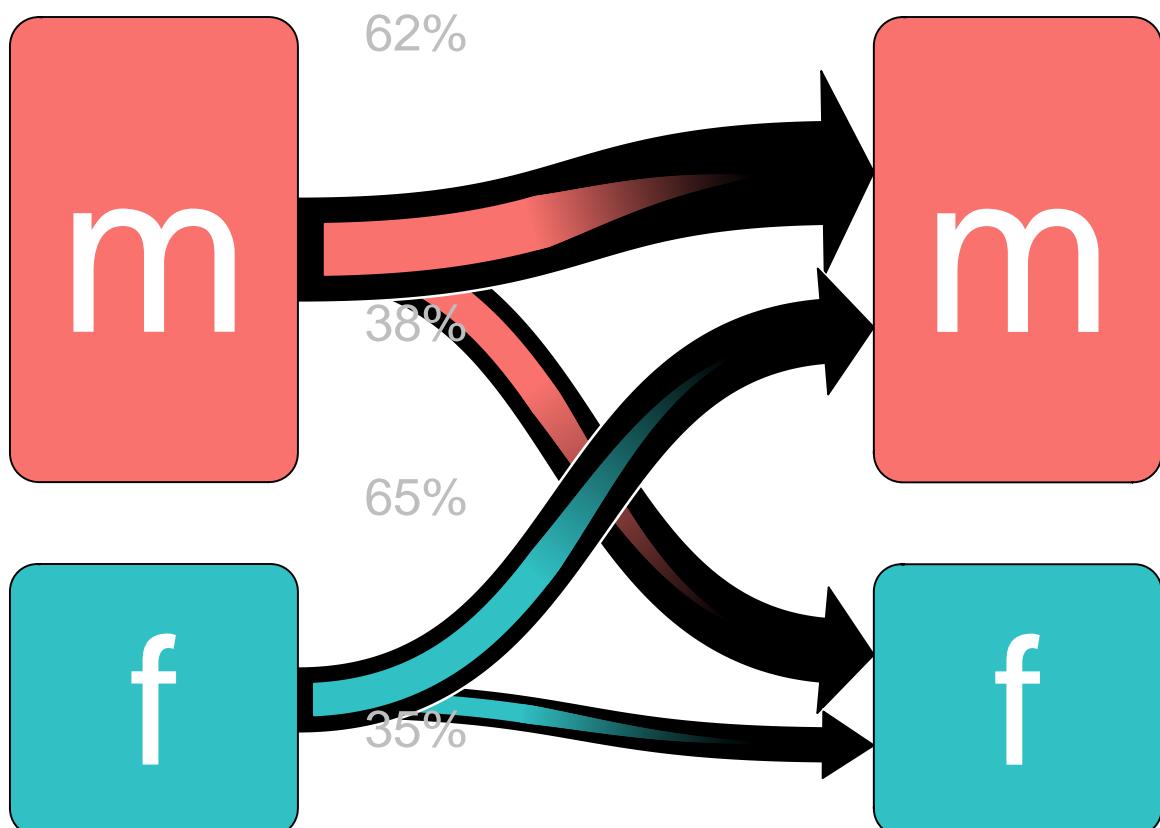
Previous Speaker Lightning Returns Final Fantasy XIII Next Speaker
Expected



Previous
Speaker

Next
Speaker

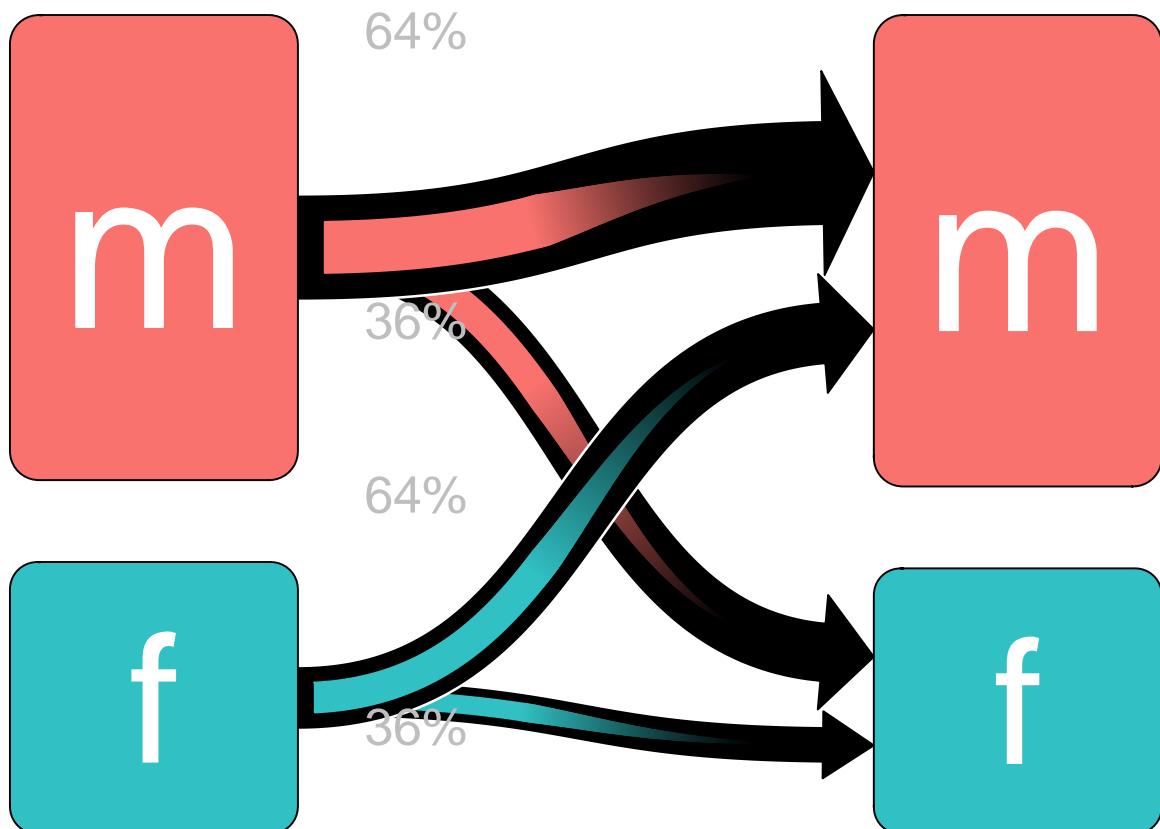
Final Fantasy XIV
Empirical



Previous
Speaker

Final Fantasy XIV
Expected

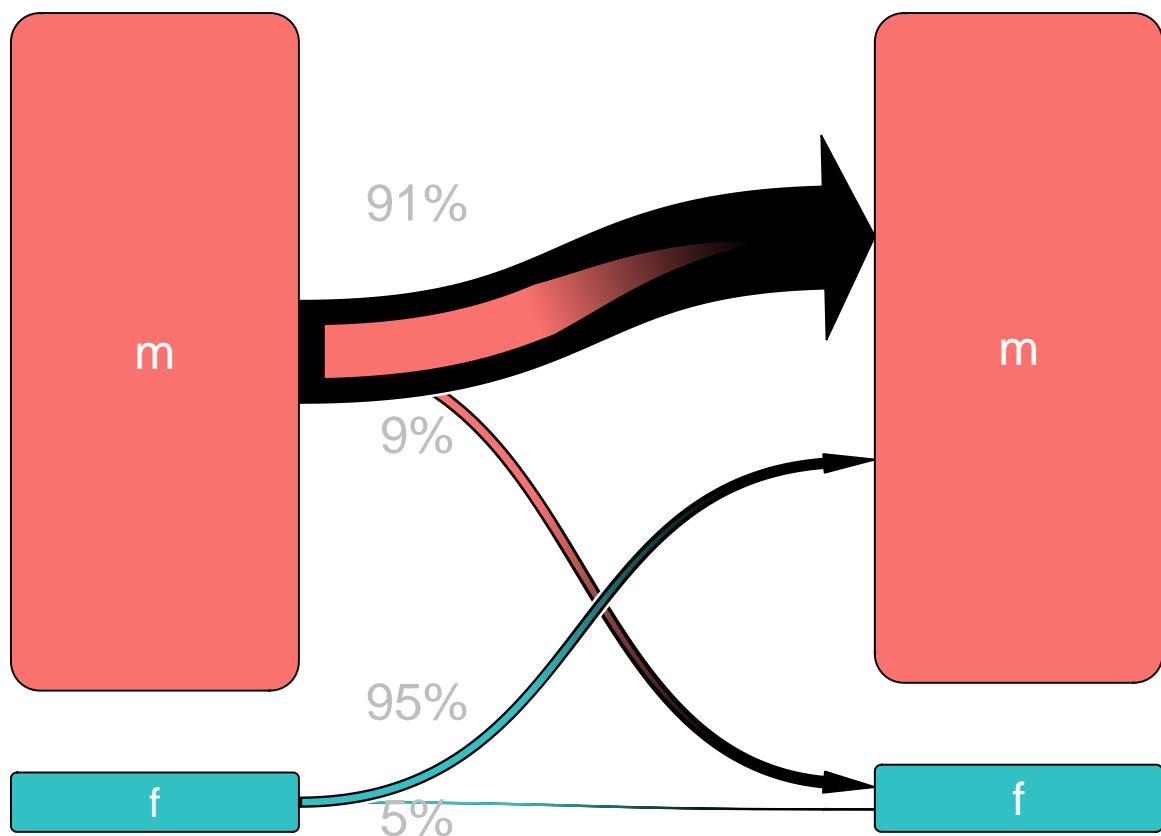
Next
Speaker



Previous Speaker

Next Speaker

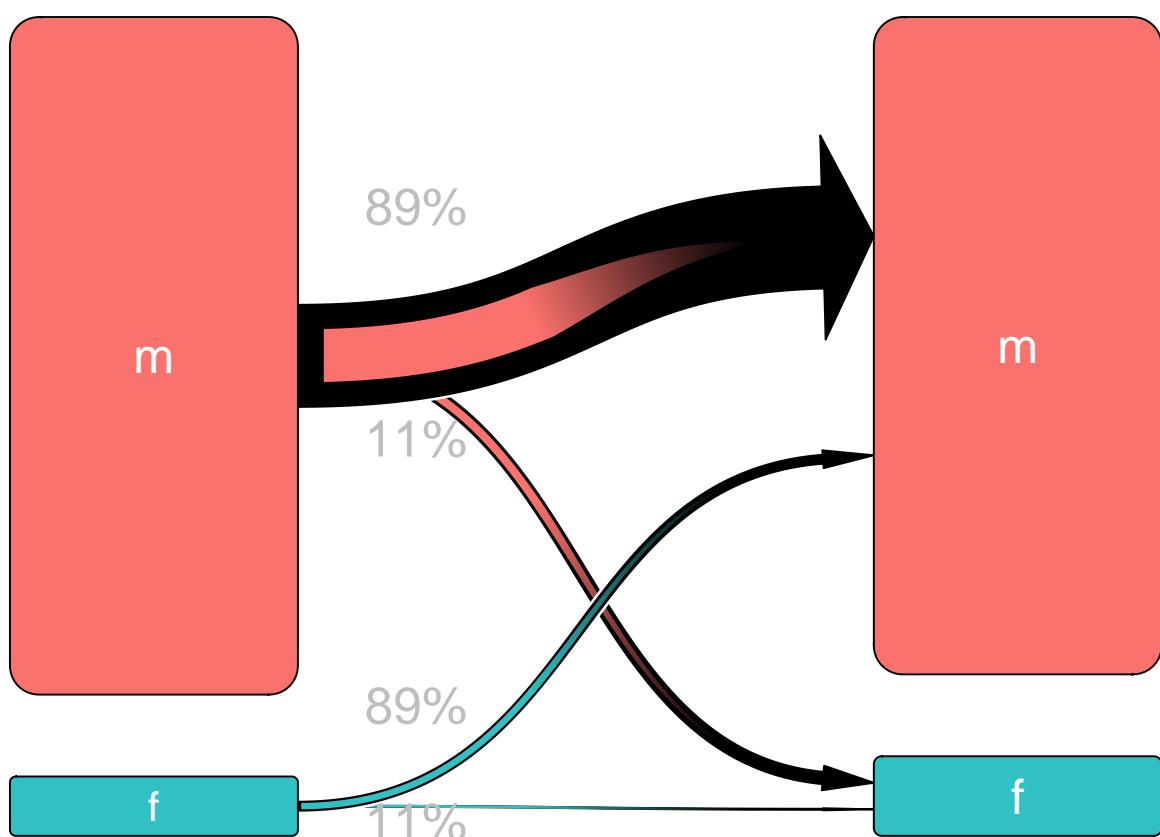
FF XV
Empirical



Previous
Speaker

Next
Speaker

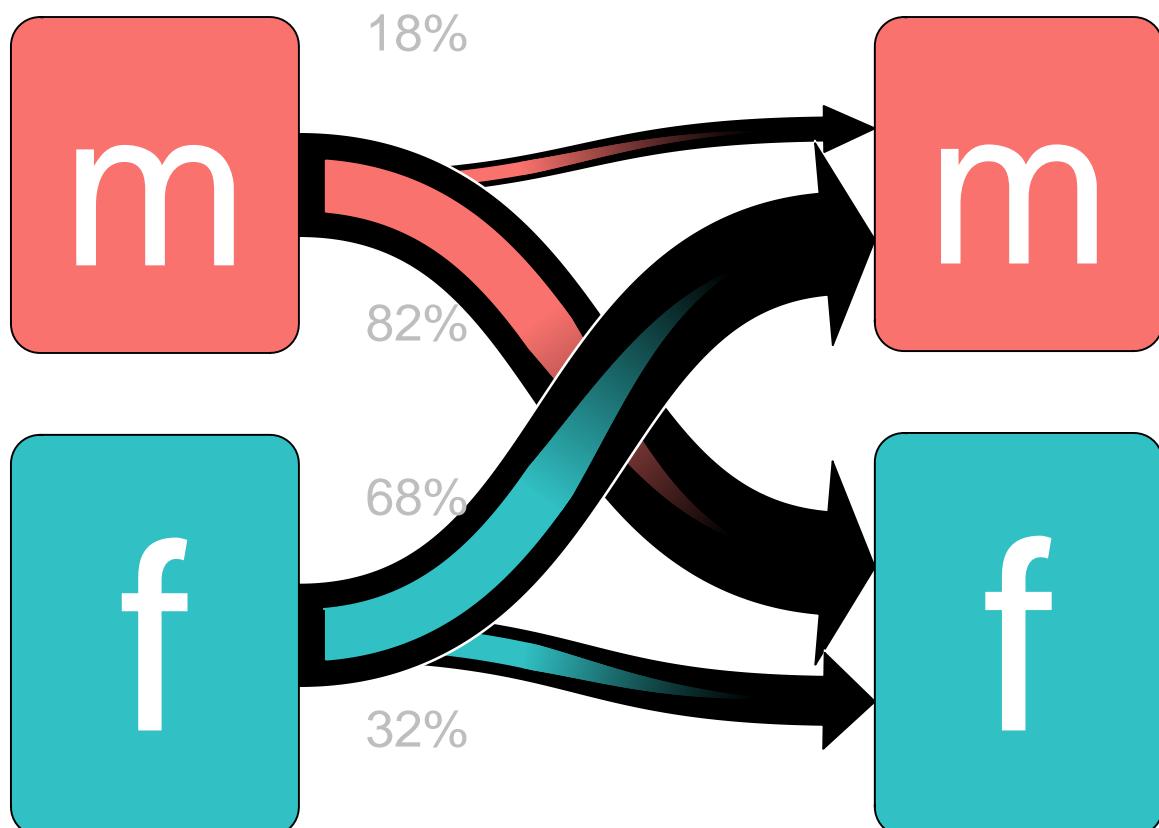
FF XV
Expected



Previous
Speaker

Horizon Zero Dawn
Empirical

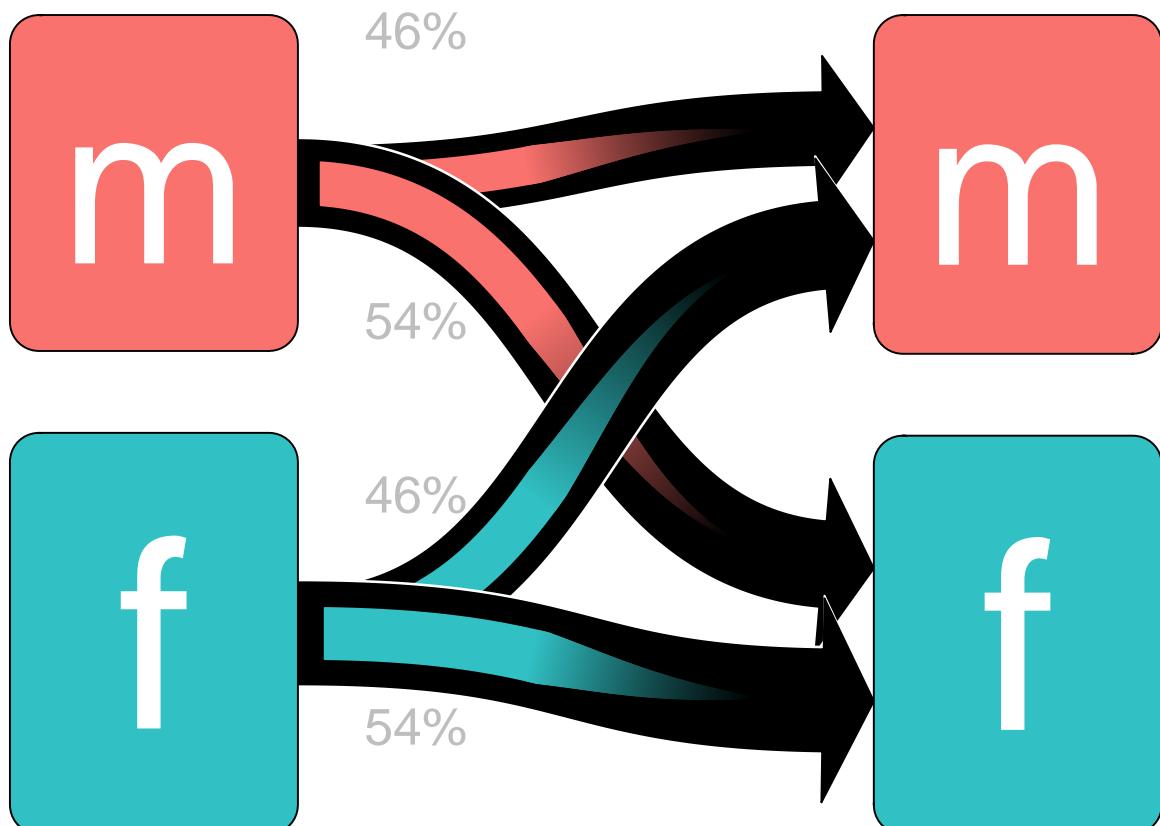
Next
Speaker



Previous
Speaker

Horizon Zero Dawn
Expected

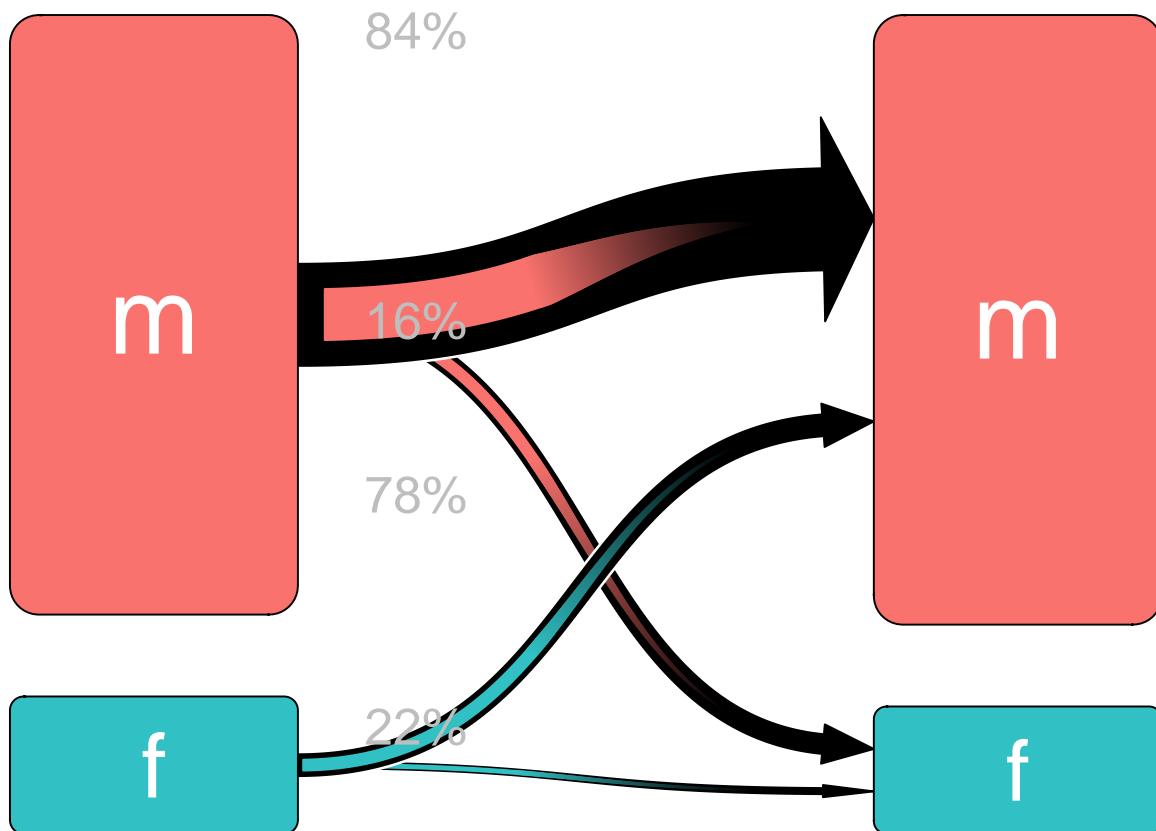
Next
Speaker



Previous Speaker

Kingdom Hearts
Empirical

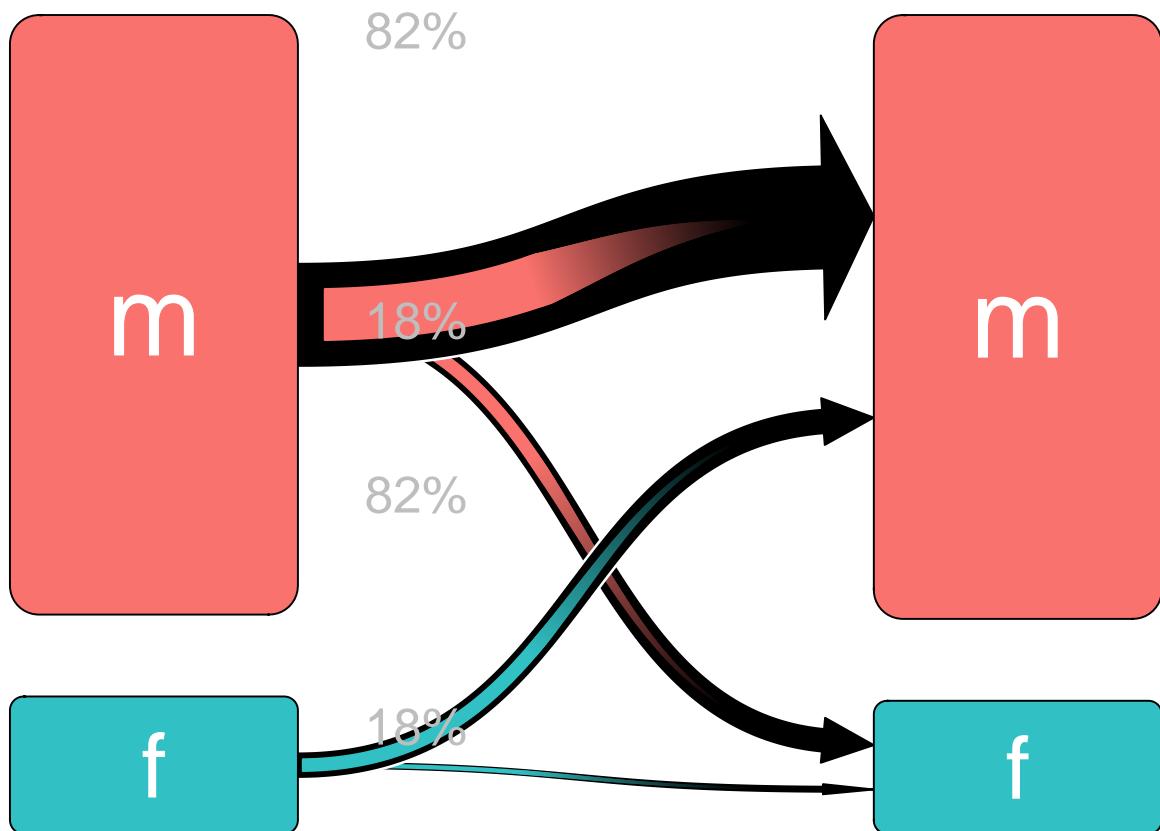
Next Speaker



Previous
Speaker

Kingdom Hearts
Expected

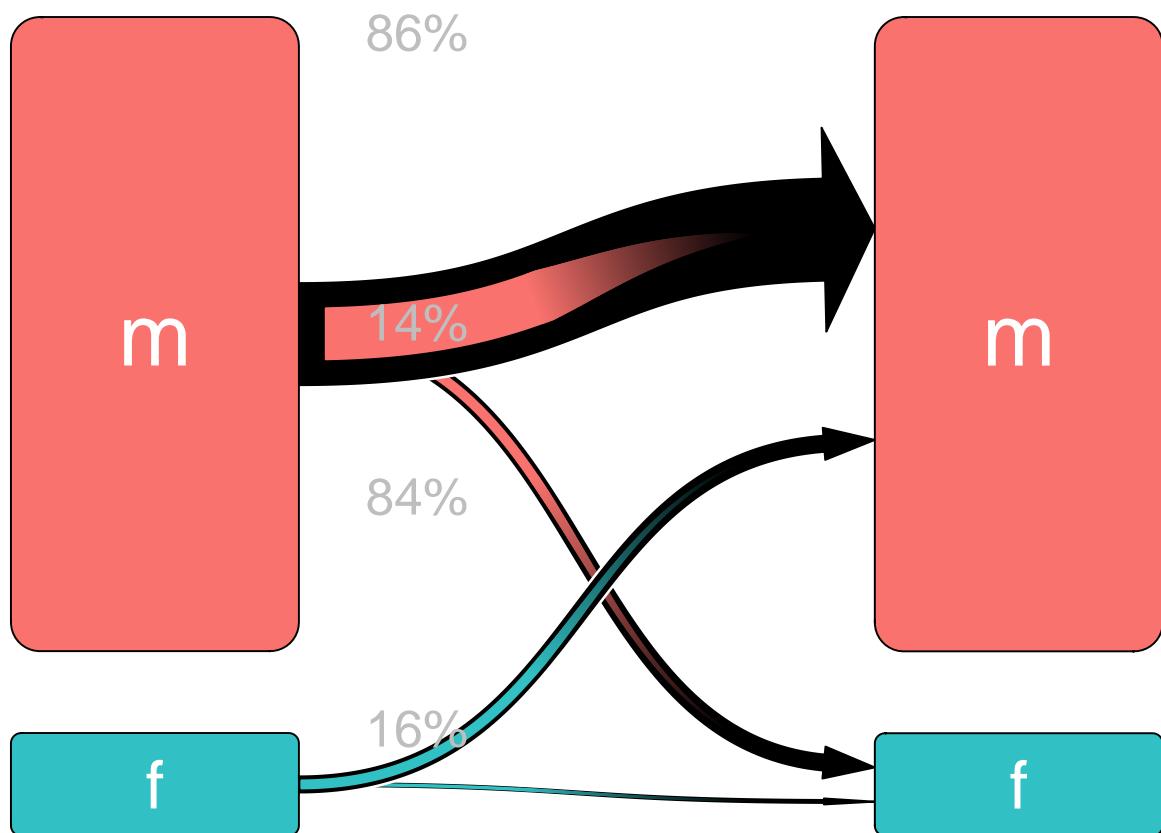
Next
Speaker



Previous Speaker

Kingdom Hearts II
Empirical

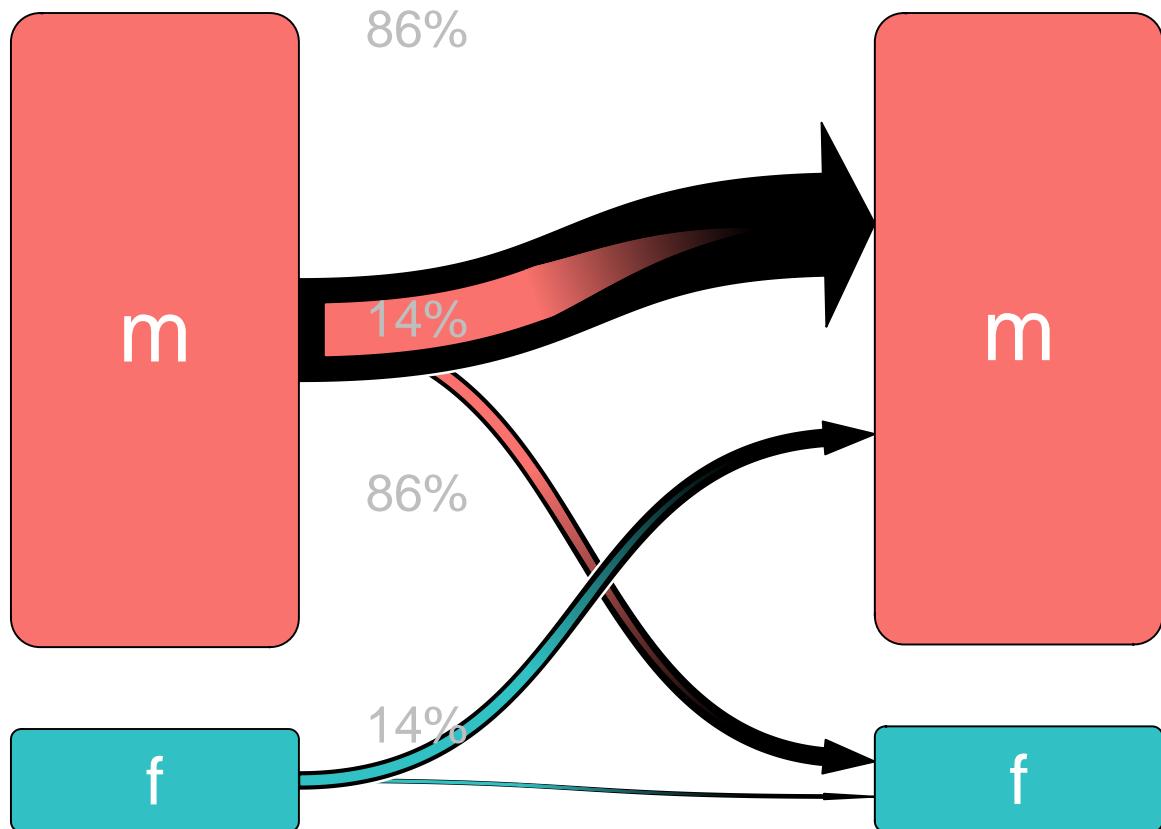
Next Speaker



Previous
Speaker

Kingdom Hearts II
Expected

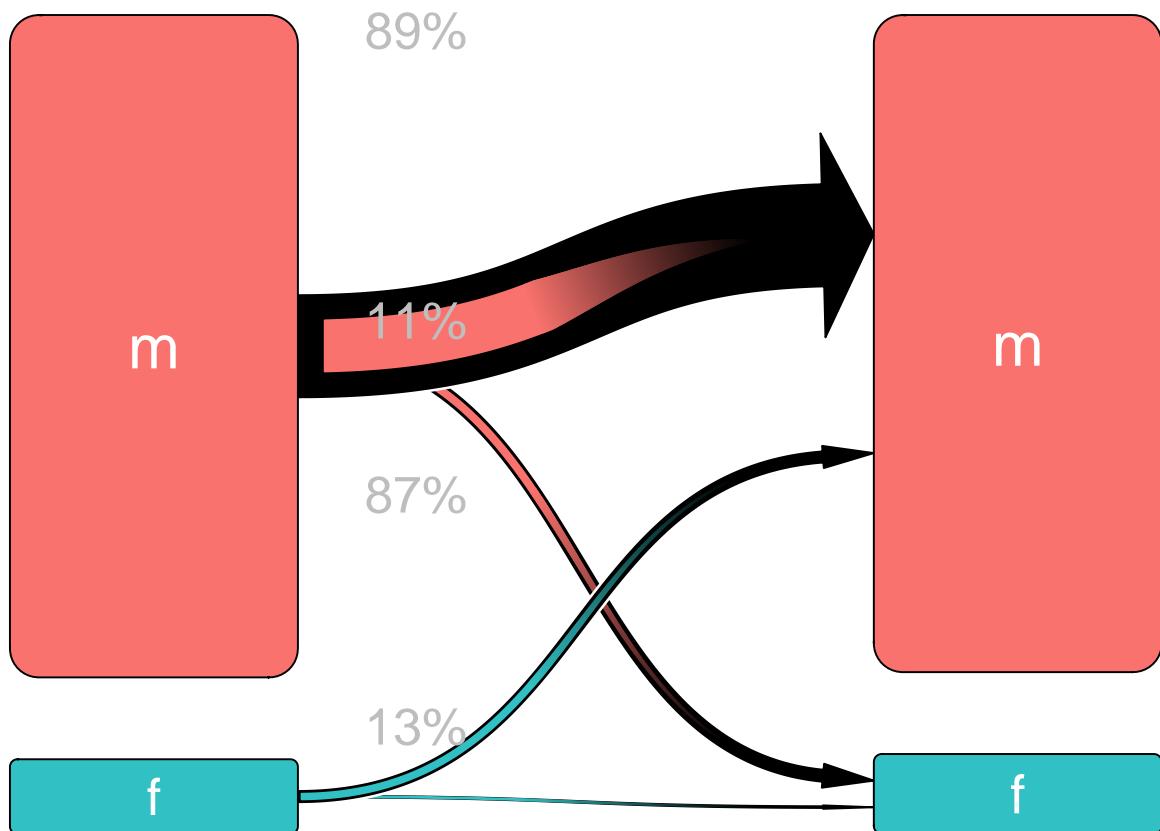
Next
Speaker



Previous Speaker

Kingdom Hearts III
Empirical

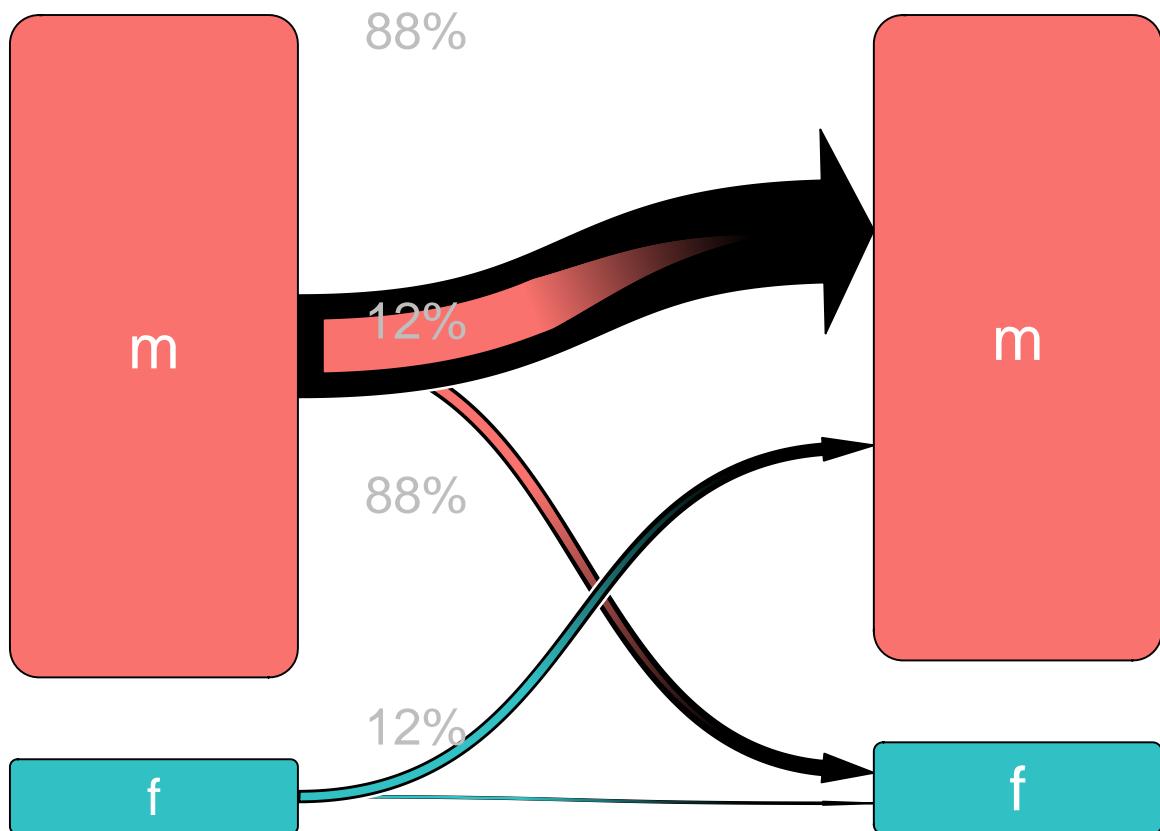
Next Speaker



Previous
Speaker

Kingdom Hearts III
Expected

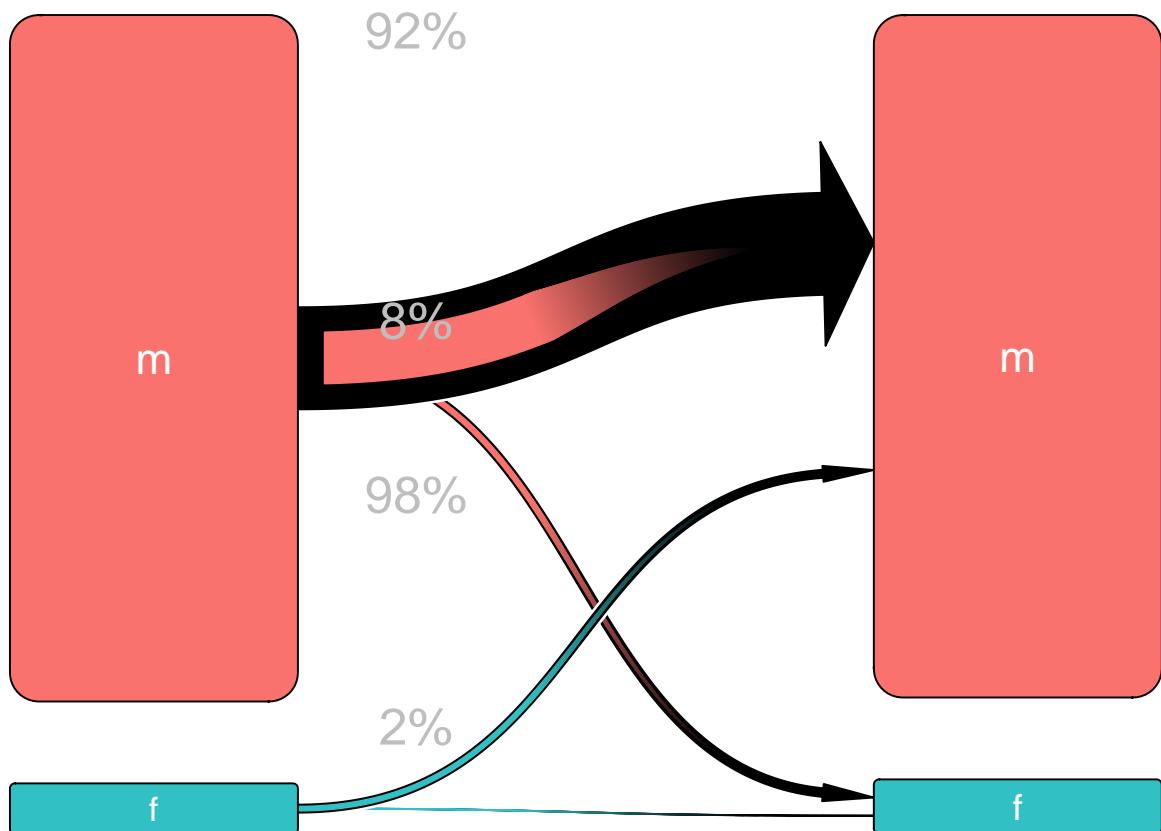
Next
Speaker



Previous Speaker

Next Speaker

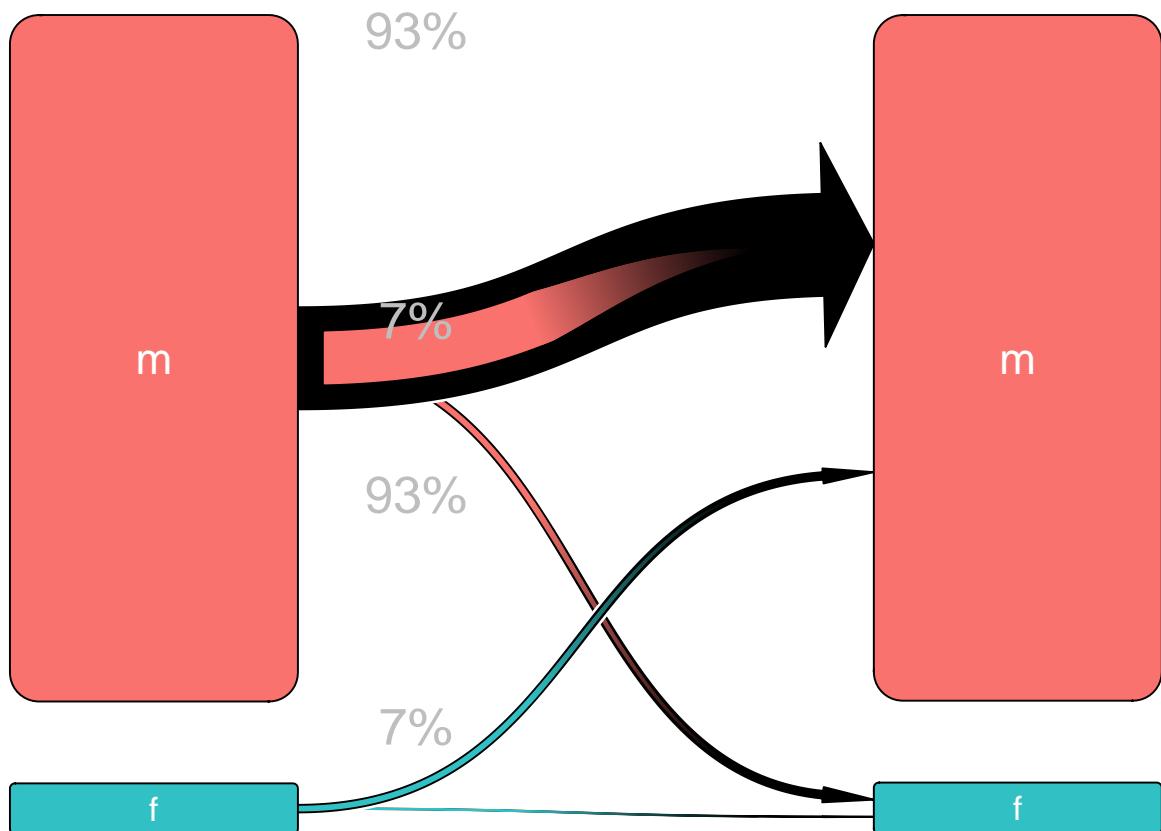
Kingdom Hearts 3D Dream Drop Distance
Empirical



Previous Speaker

Next Speaker

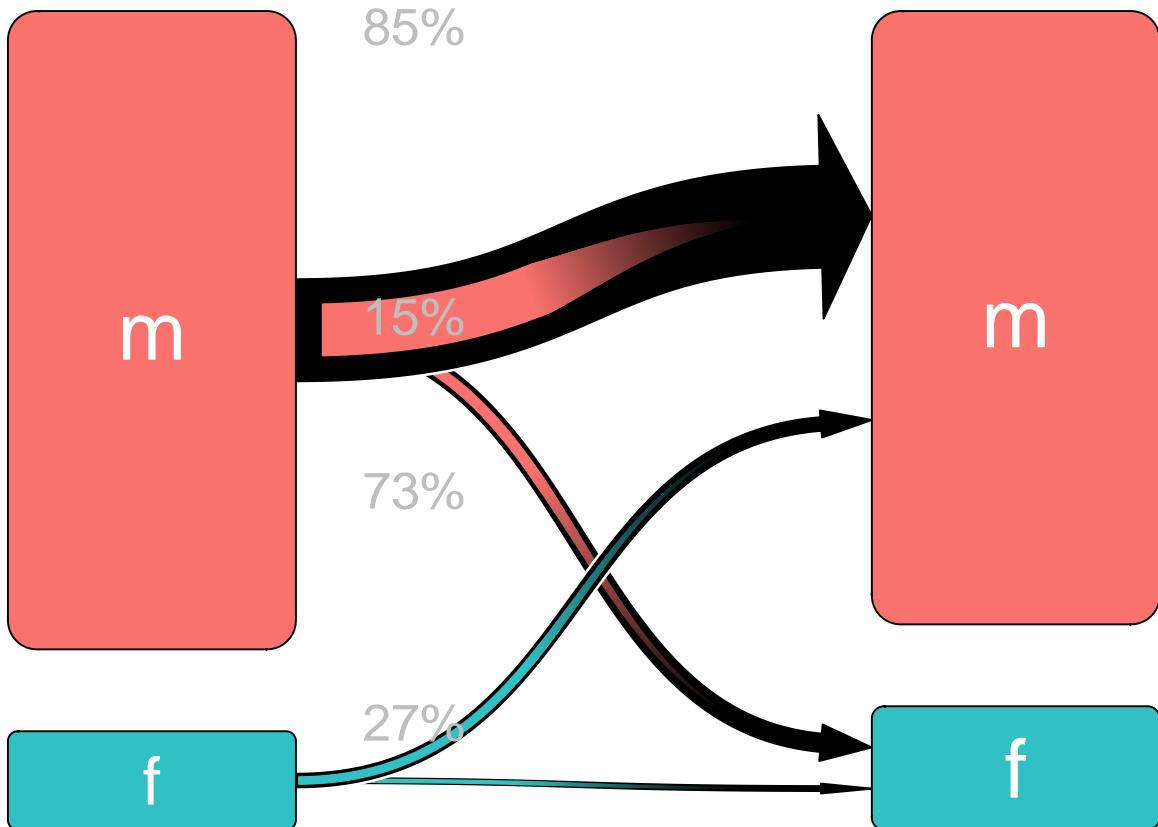
Kingdom Hearts 3D Dream Drop Distance
Expected



Previous Speaker

Kings Quest VI
Empirical

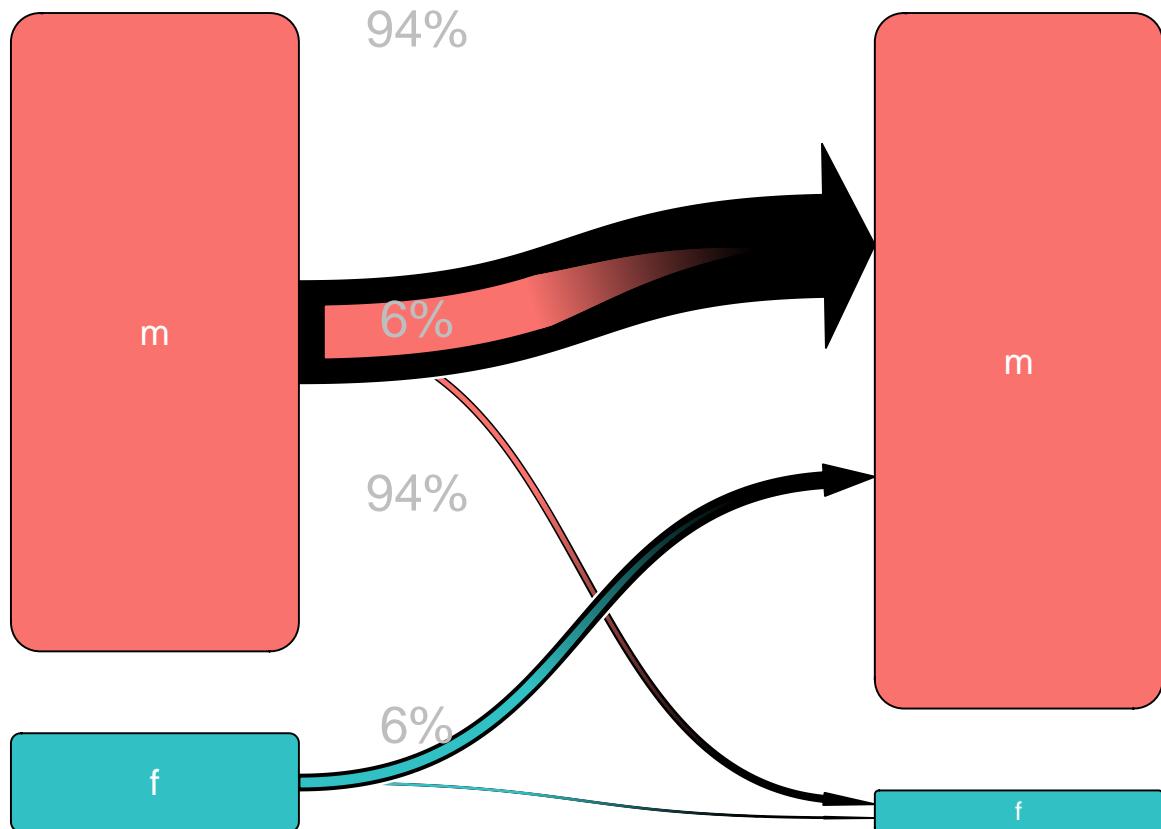
Next Speaker



Previous Speaker

Kings Quest VI
Expected

Next Speaker



Previous

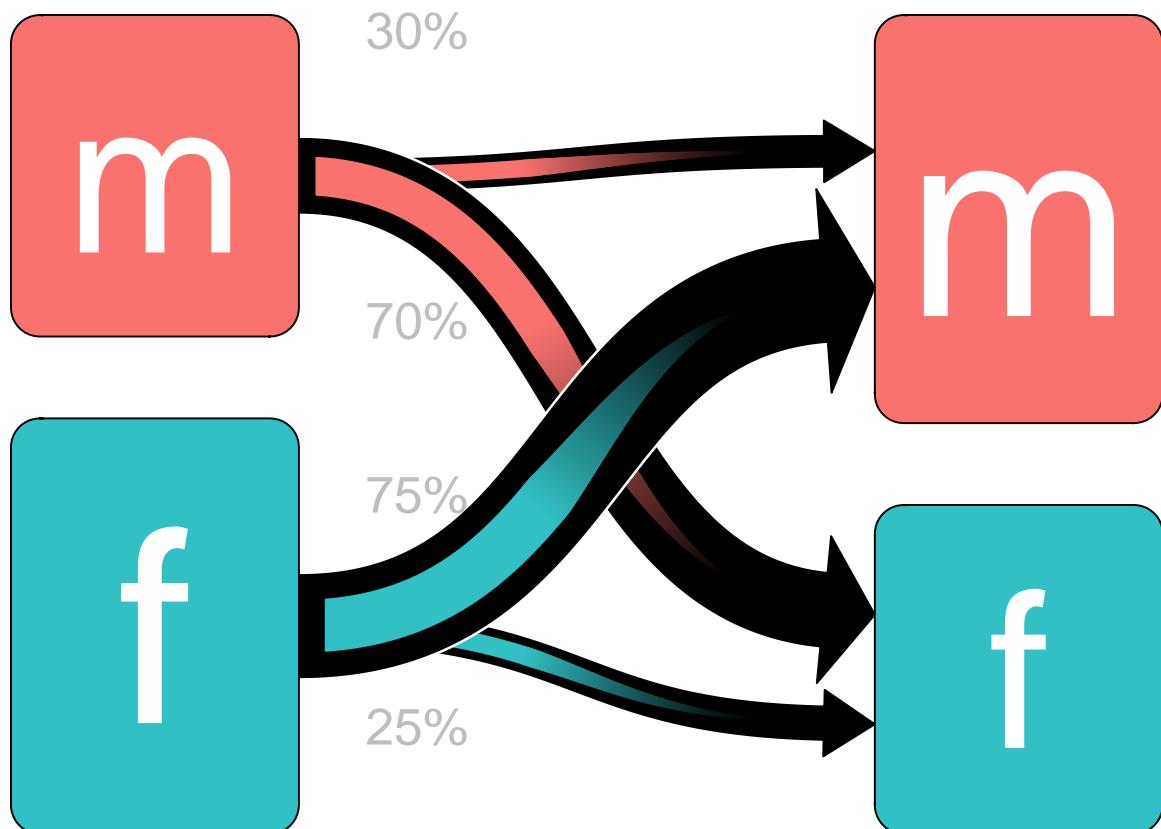
Speaker

Kings Quest VII The Princeless Bride

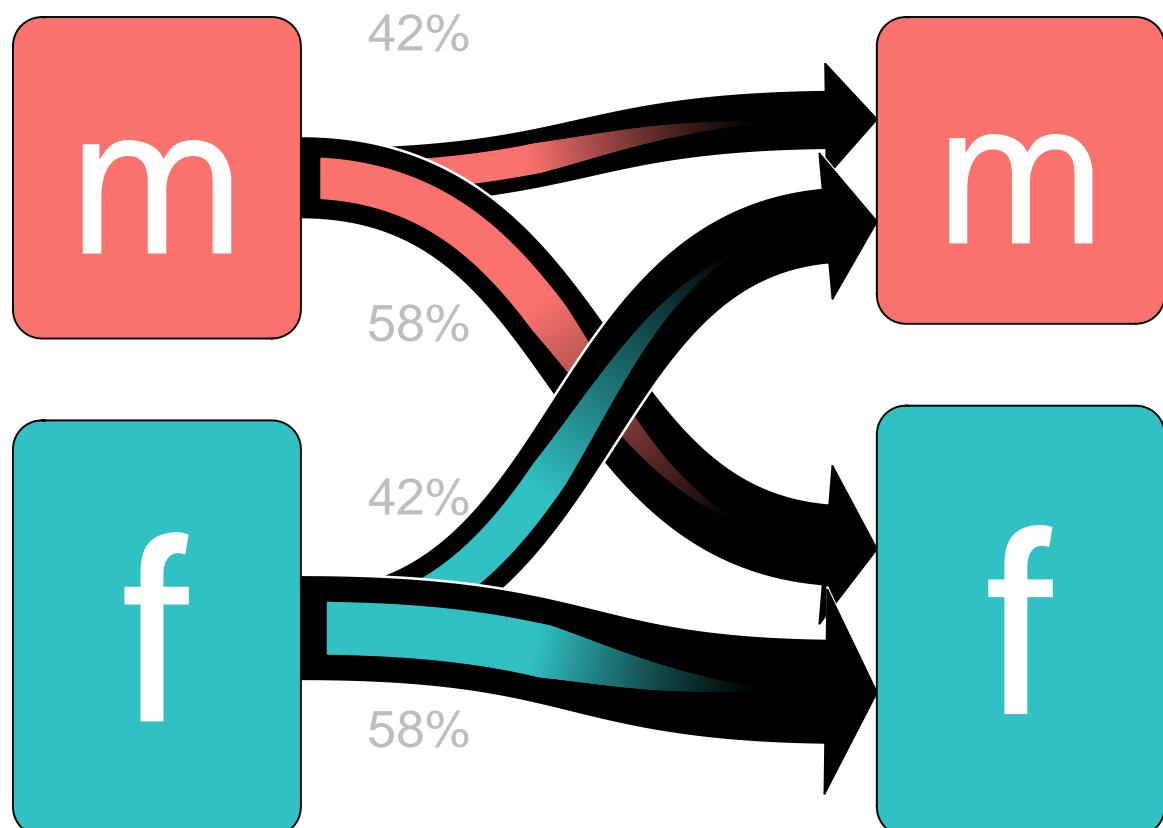
Empirical

Next

Speaker



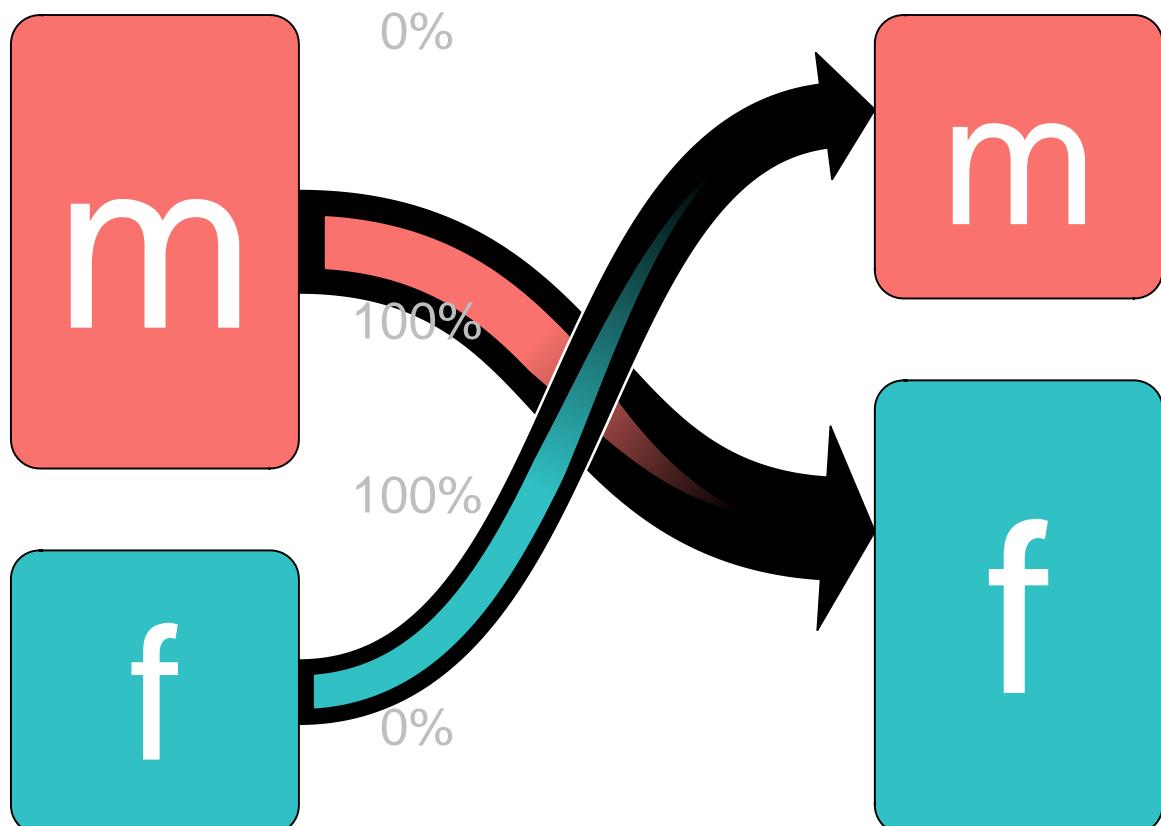
Previous Speaker Next Speaker
Kings Quest VII The Princeless Bride
Expected



Previous
Speaker

Next
Speaker

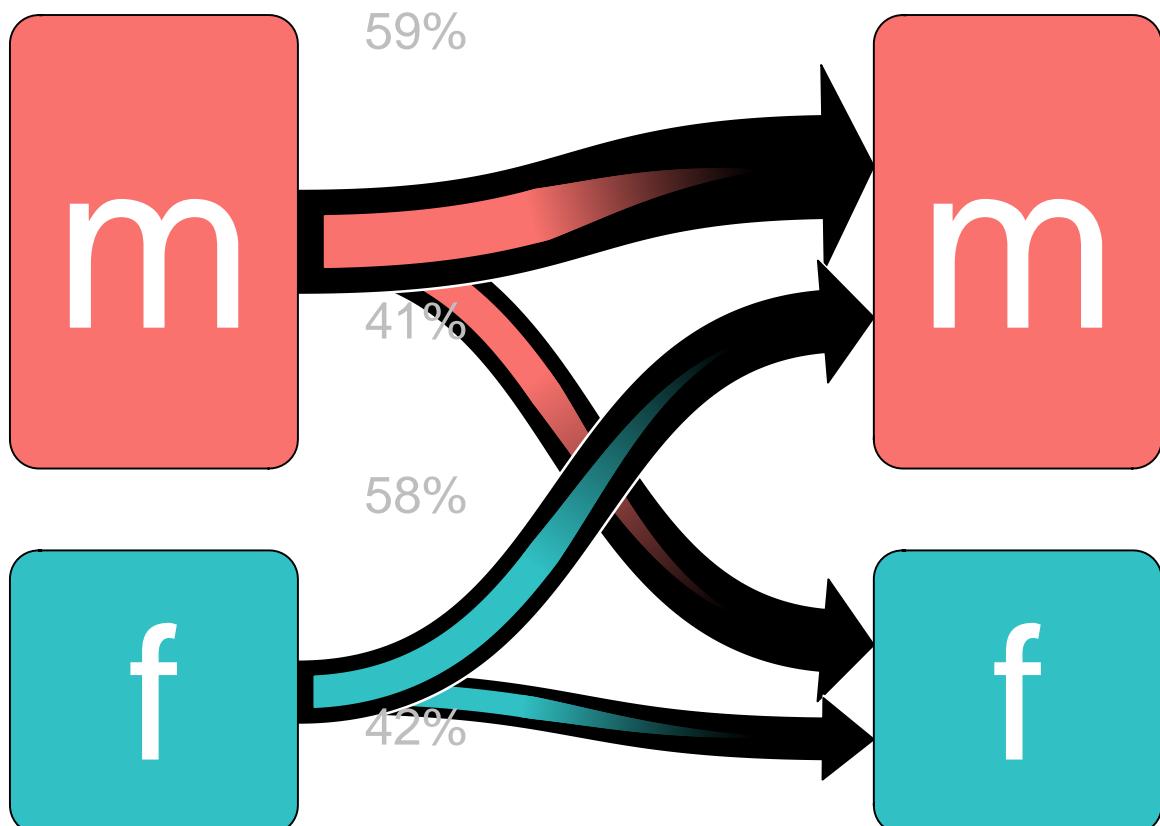
Mass Effect 2
Empirical



Previous
Speaker

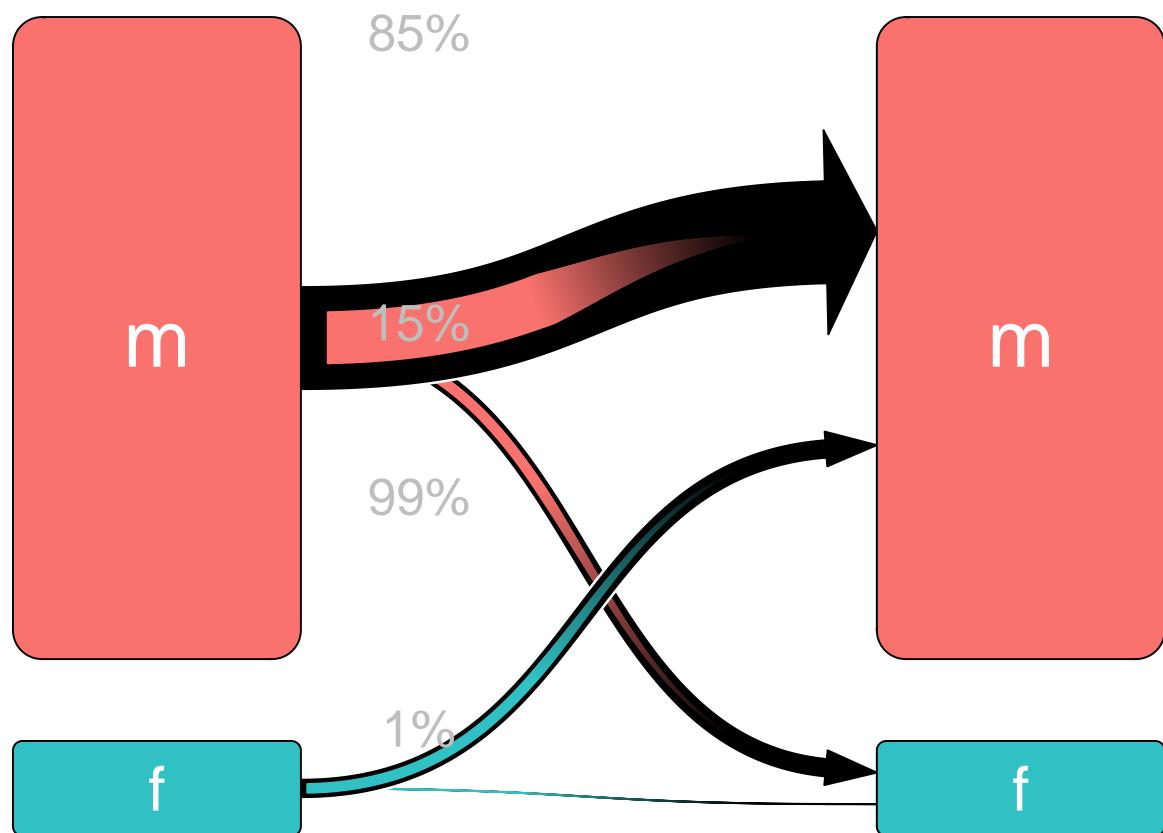
Mass Effect 2
Expected

Next
Speaker

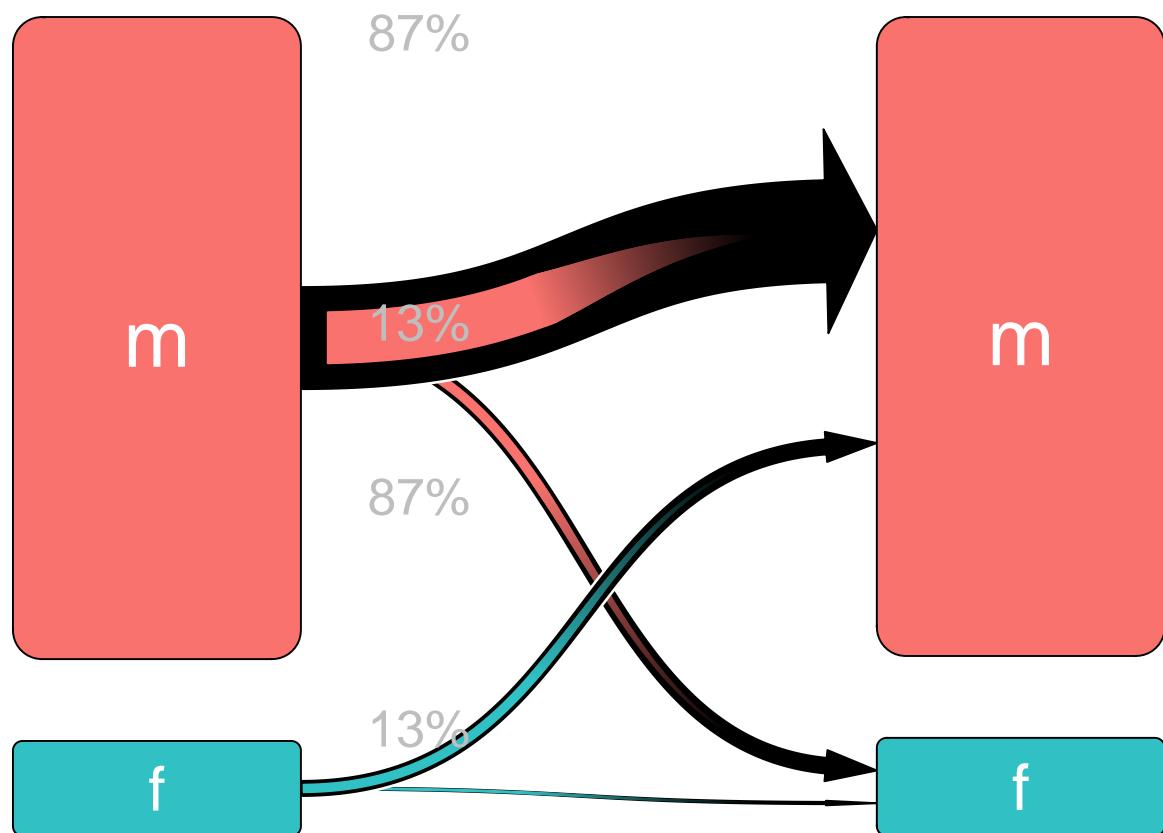


Previous Speaker Monkey Island 2 LeChucks Revenge
Next Speaker

Empirical



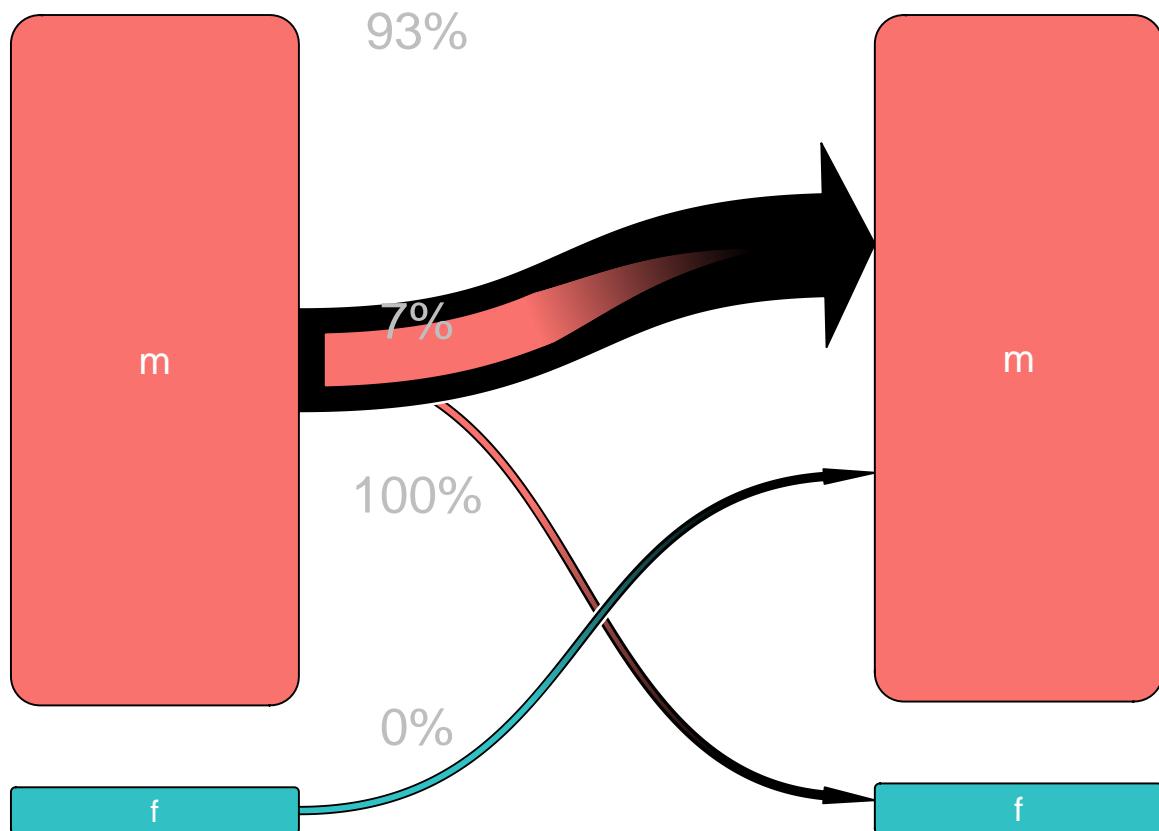
Previous Speaker Monkey Island 2 LeChucks Revenge Next Speaker
Expected



Previous Speaker

The Curse of Monkey Island
Empirical

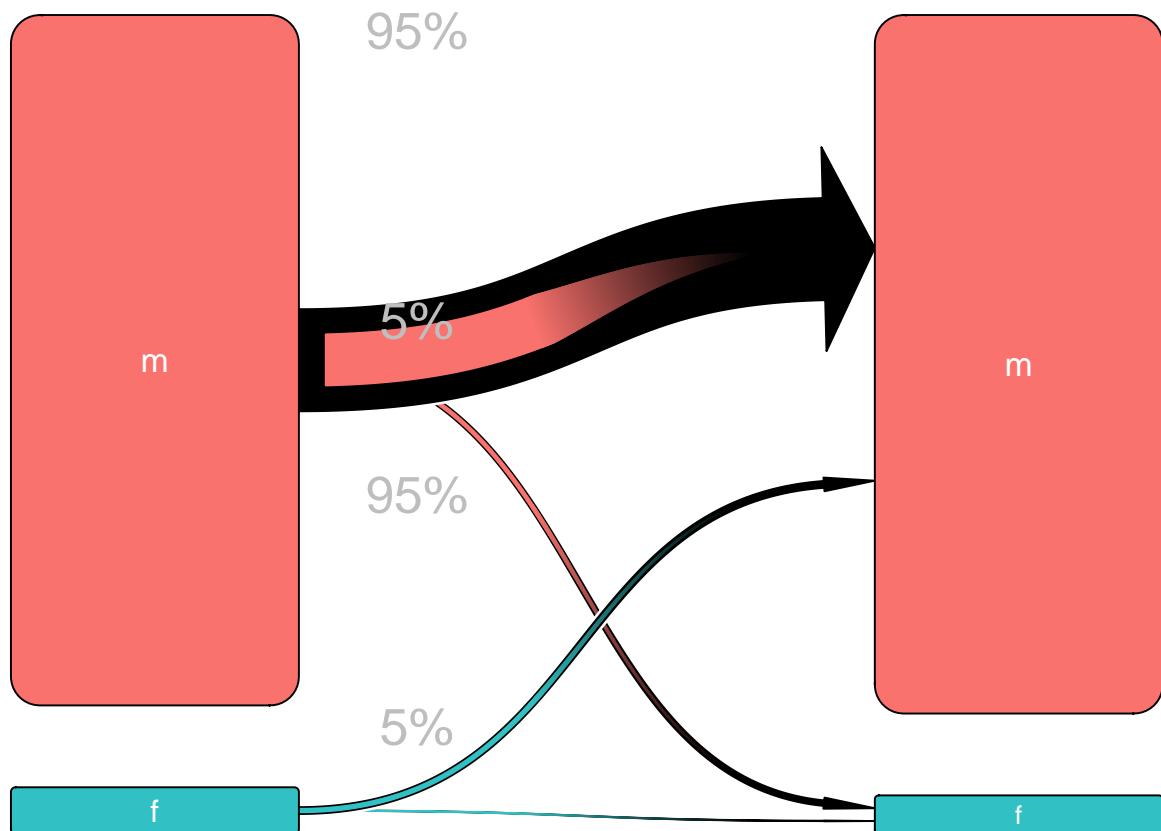
Next Speaker



Previous Speaker

The Curse of Monkey Island
Expected

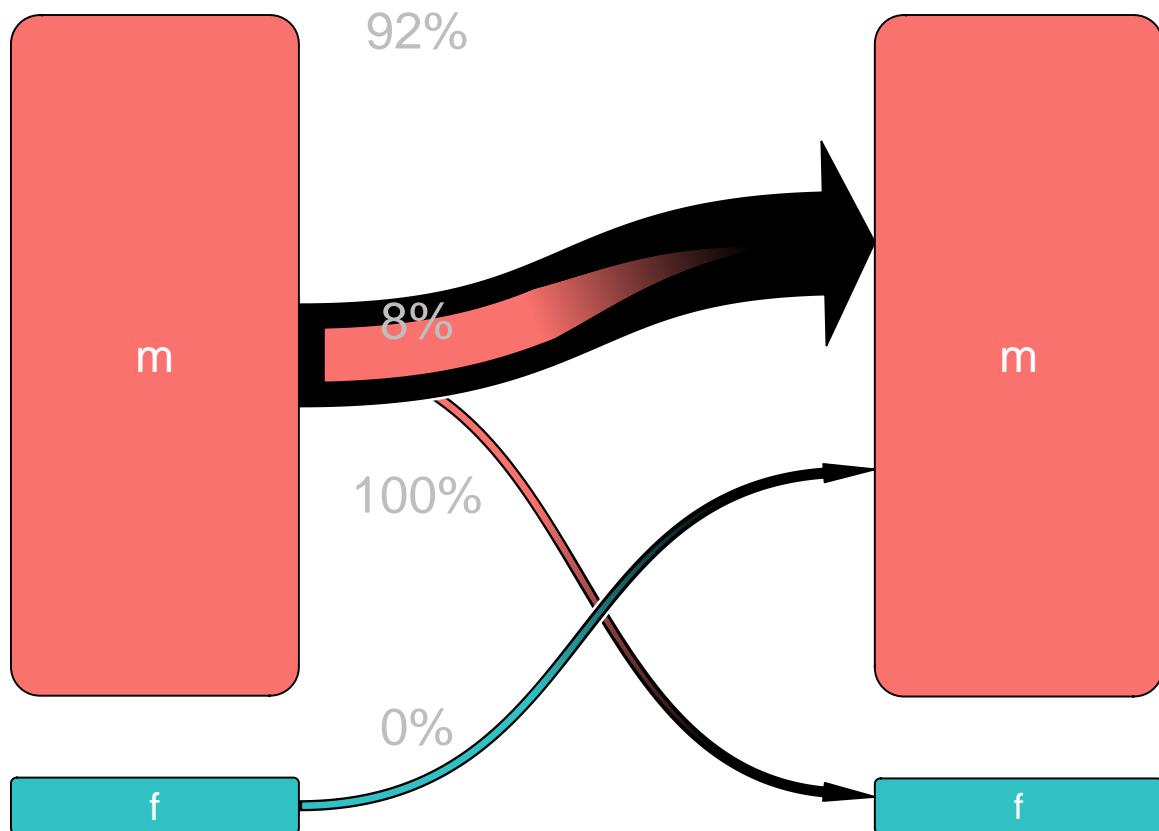
Next Speaker



Previous Speaker

The Secret of Monkey Island
Empirical

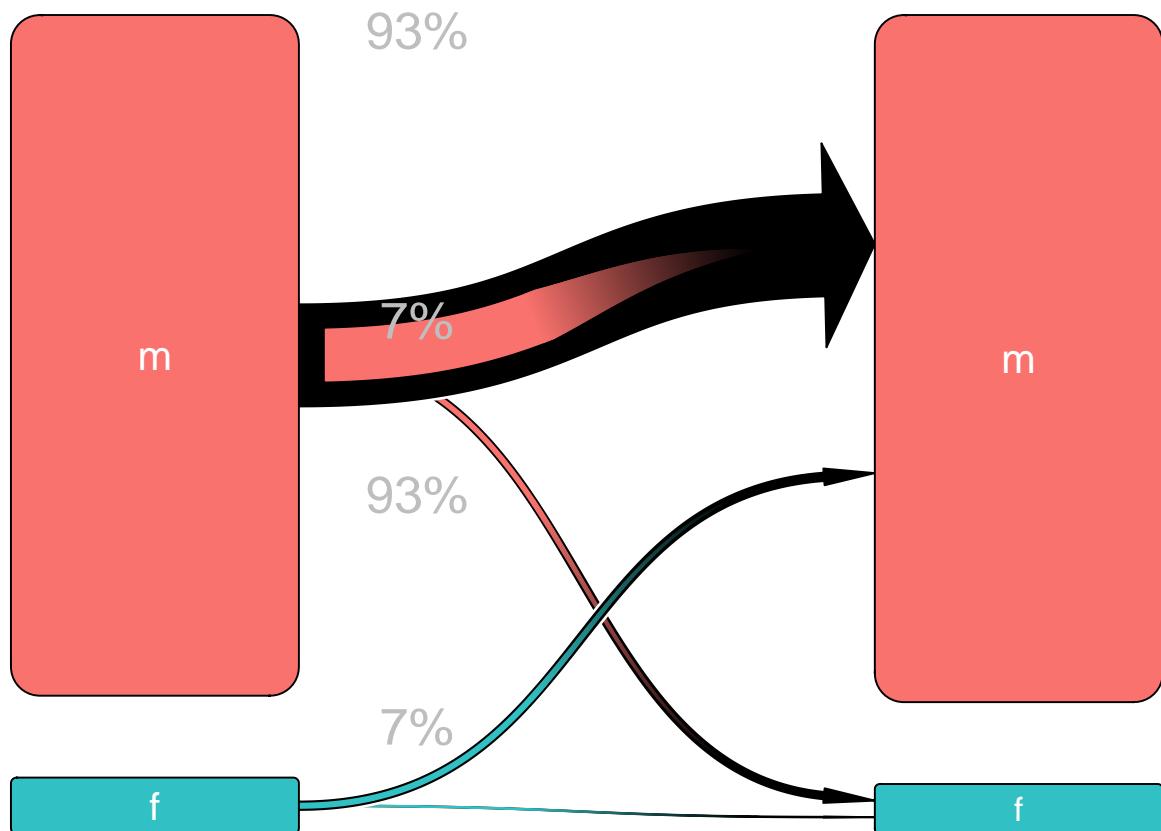
Next Speaker



Previous Speaker

The Secret of Monkey Island
Expected

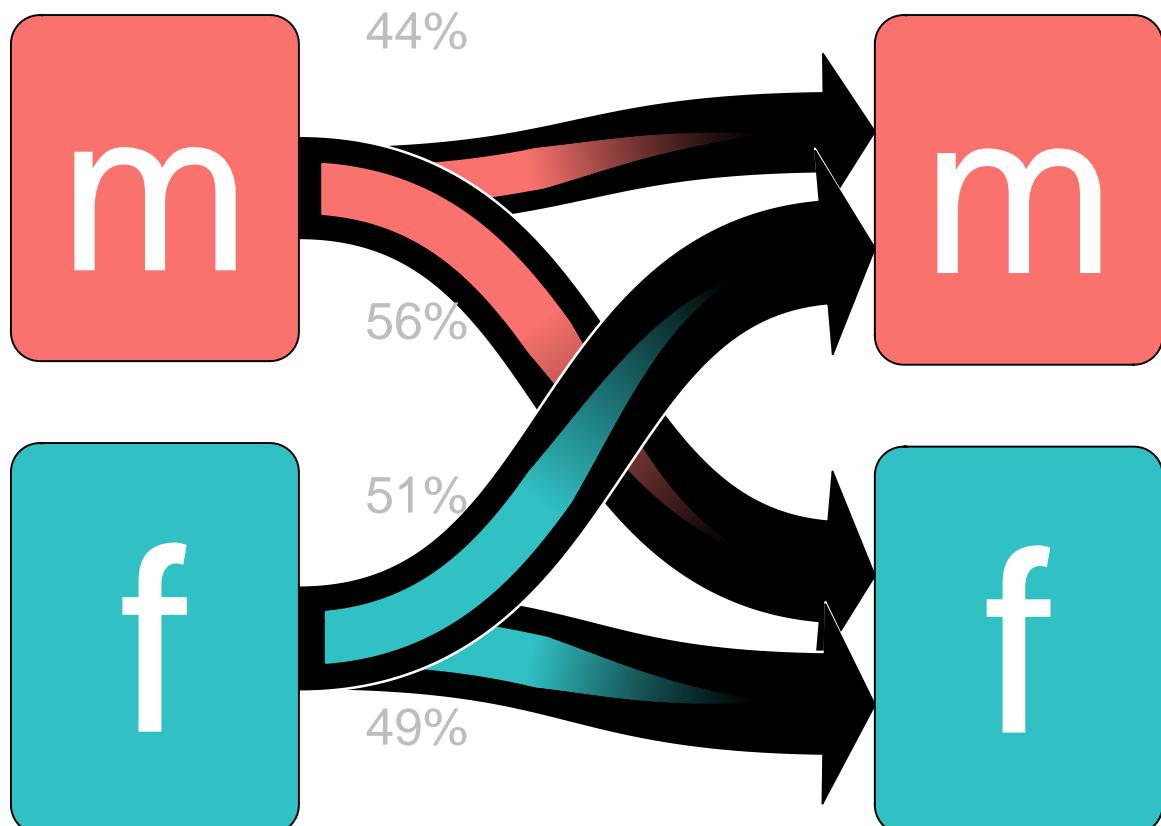
Next Speaker



Previous Speaker

Next Speaker

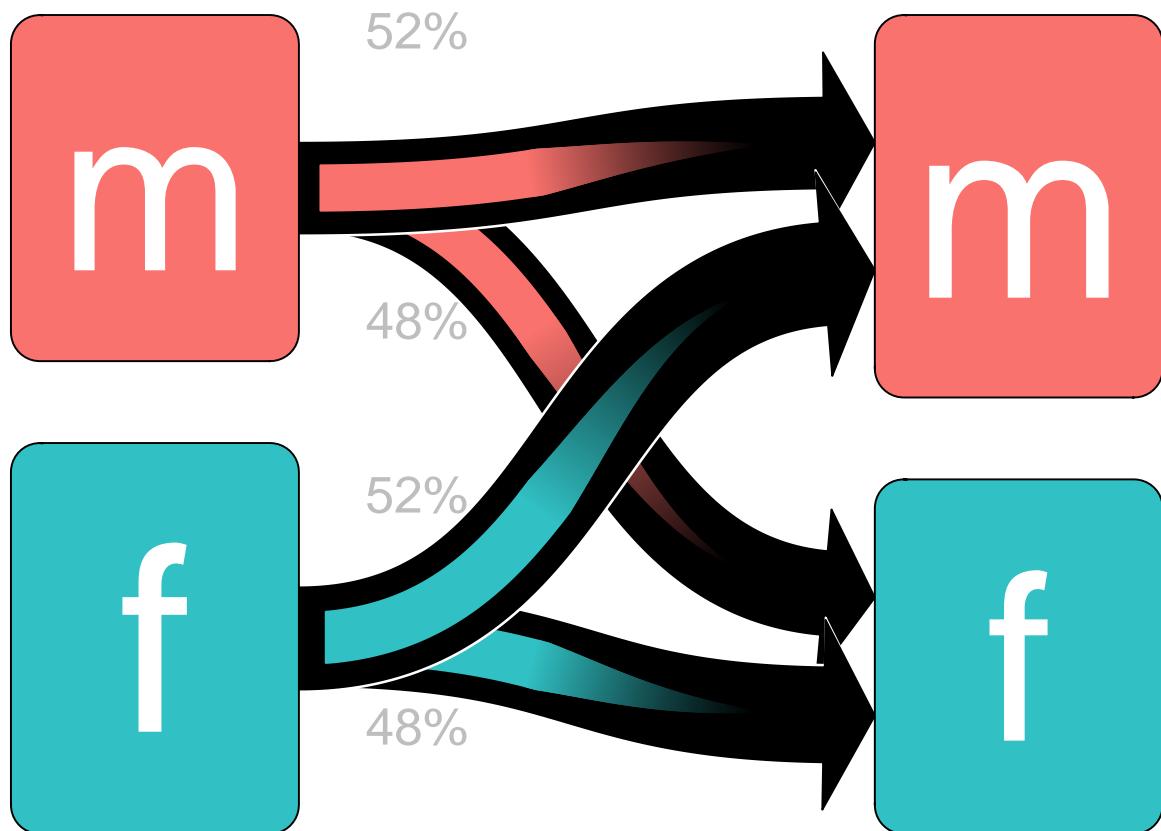
Persona 3
Empirical



Previous Speaker

Next Speaker

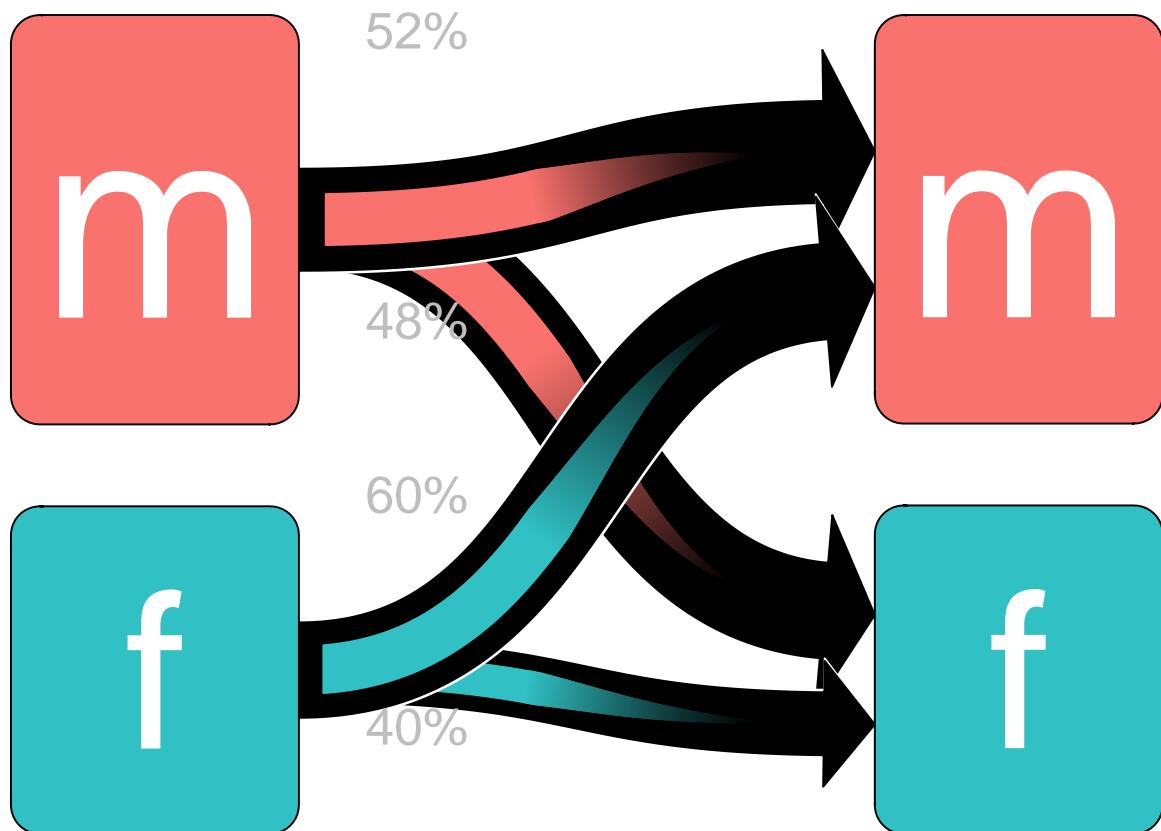
Persona 3
Expected



Previous
Speaker

Next
Speaker

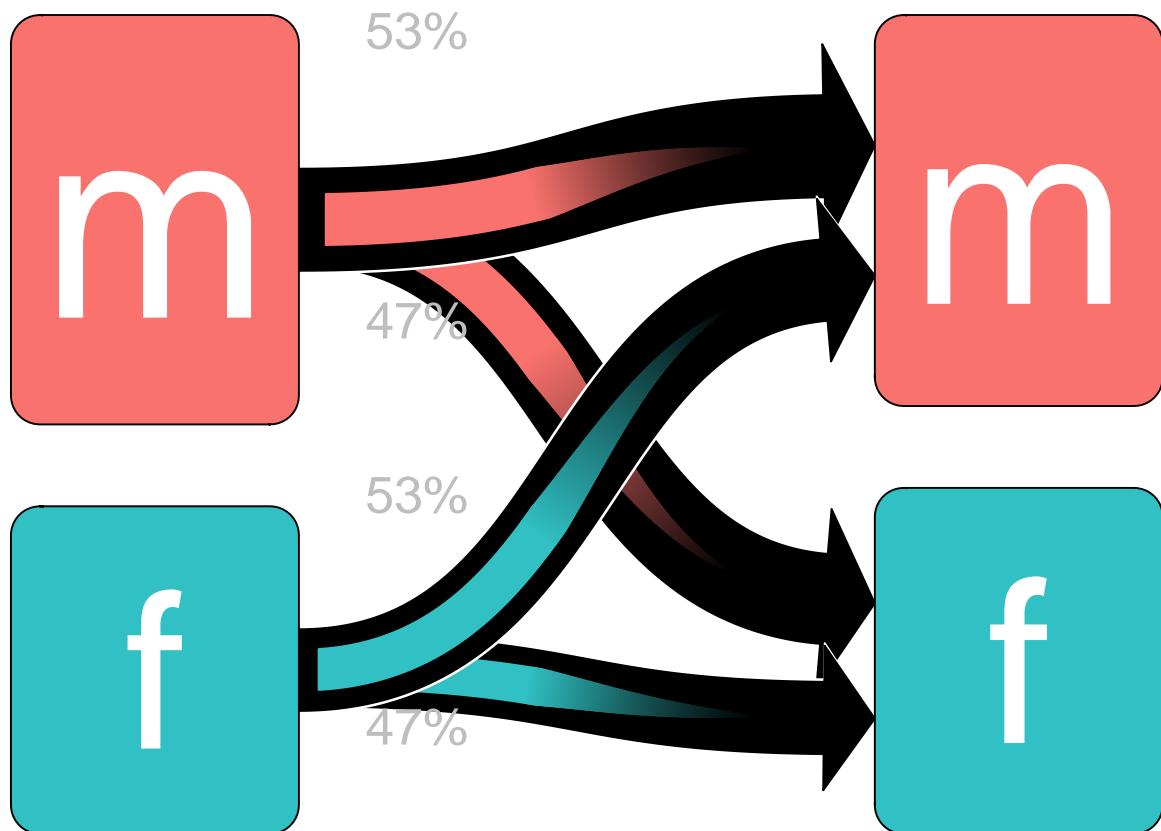
Persona 4
Empirical



Previous Speaker

Next Speaker

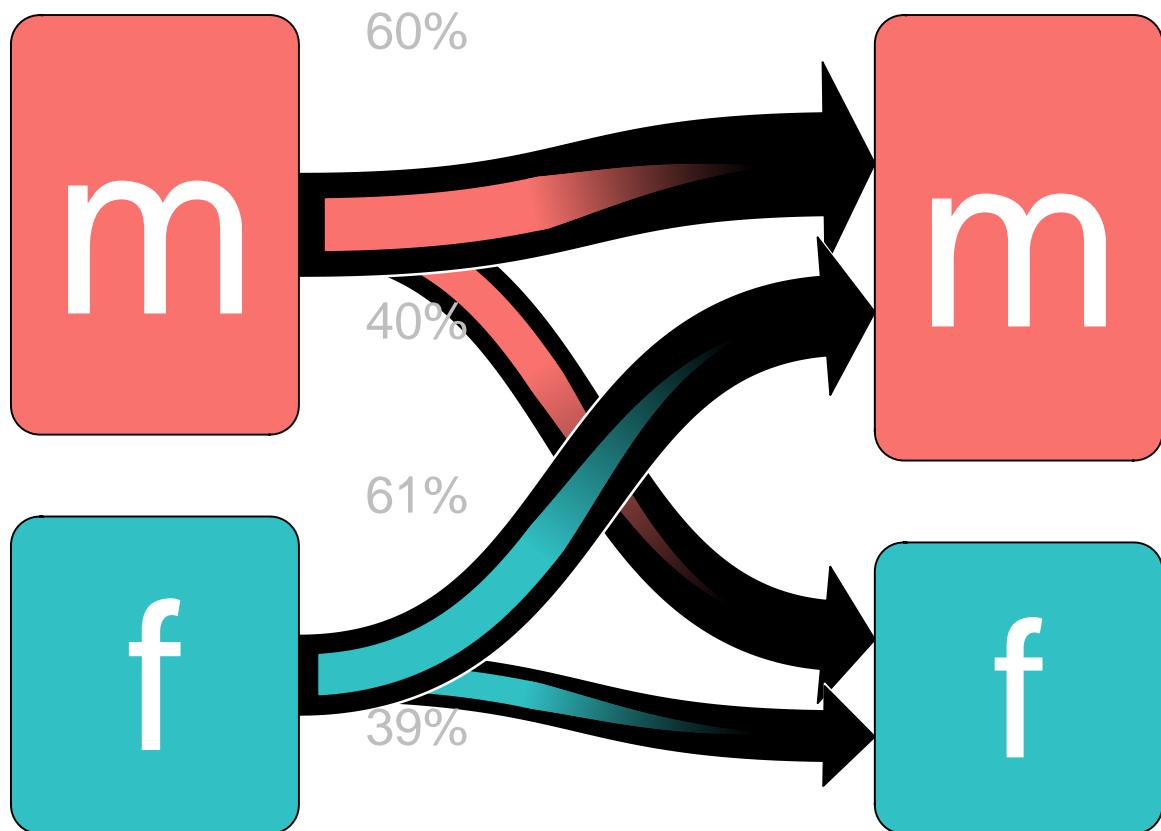
Persona 4
Expected



Previous
Speaker

Next
Speaker

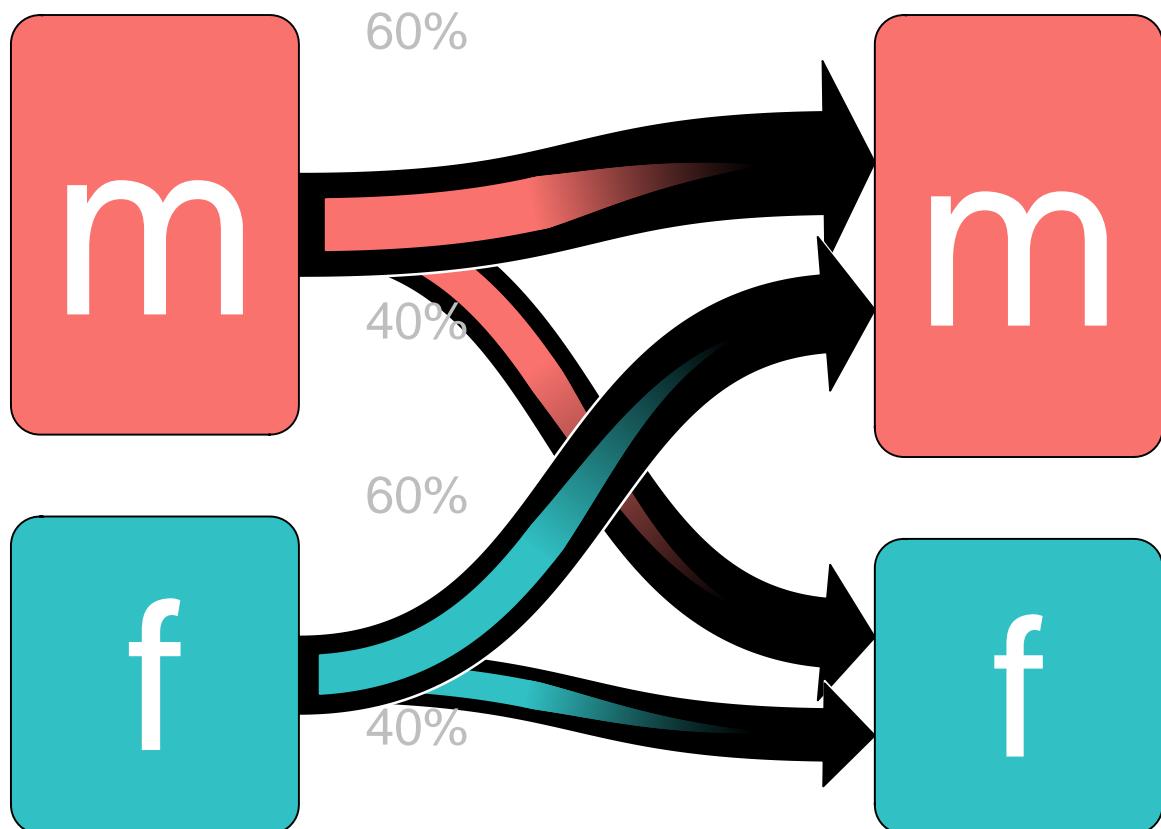
Persona 5
Empirical



Previous Speaker

Next Speaker

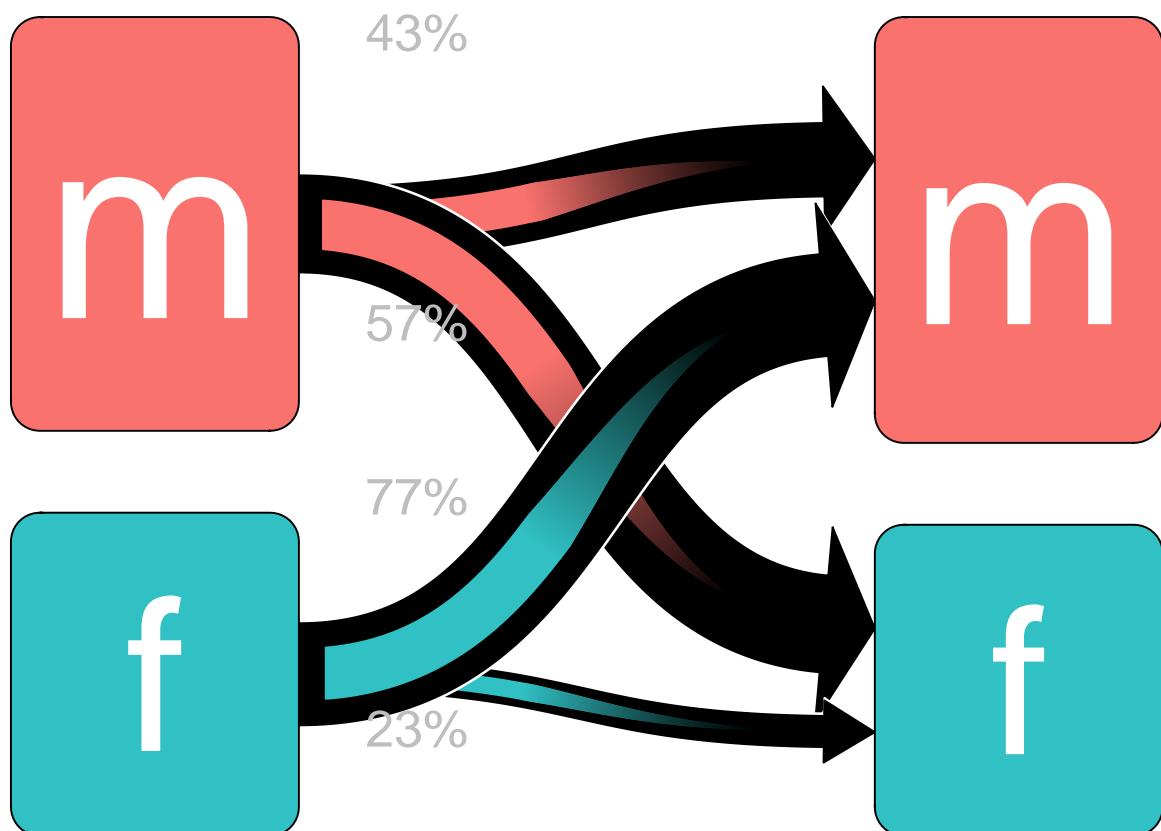
Persona 5
Expected



Previous
Speaker

Next
Speaker

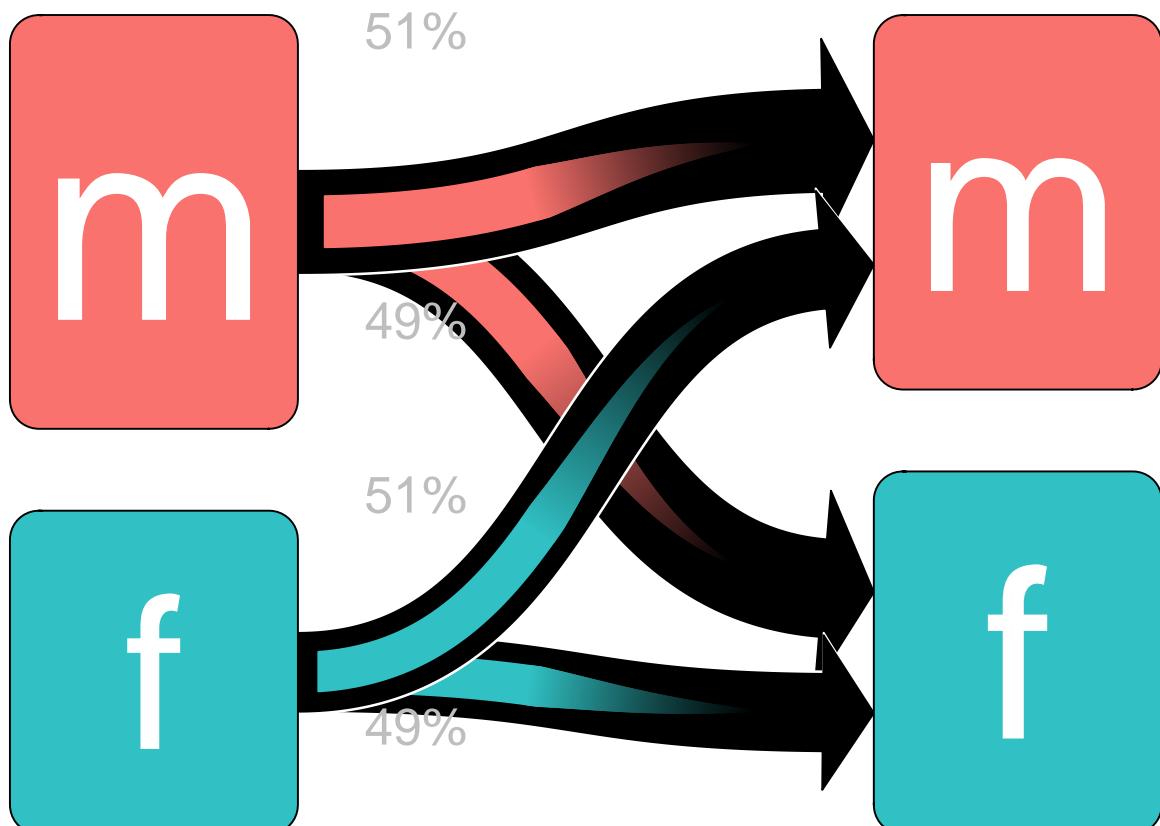
Stardew Valley
Empirical



Previous
Speaker

Next
Speaker

Stardew Valley
Expected



Previous

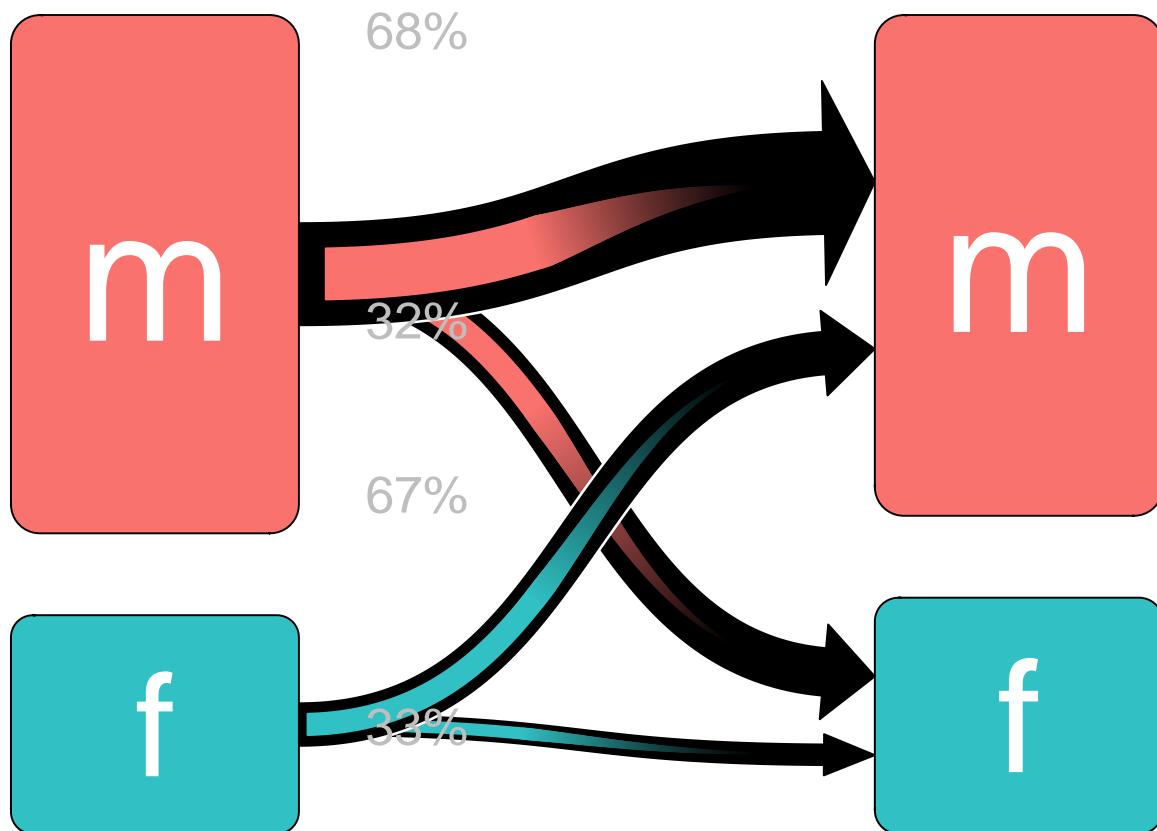
Speaker

Star Wars Knights of the Old Republic

Empirical

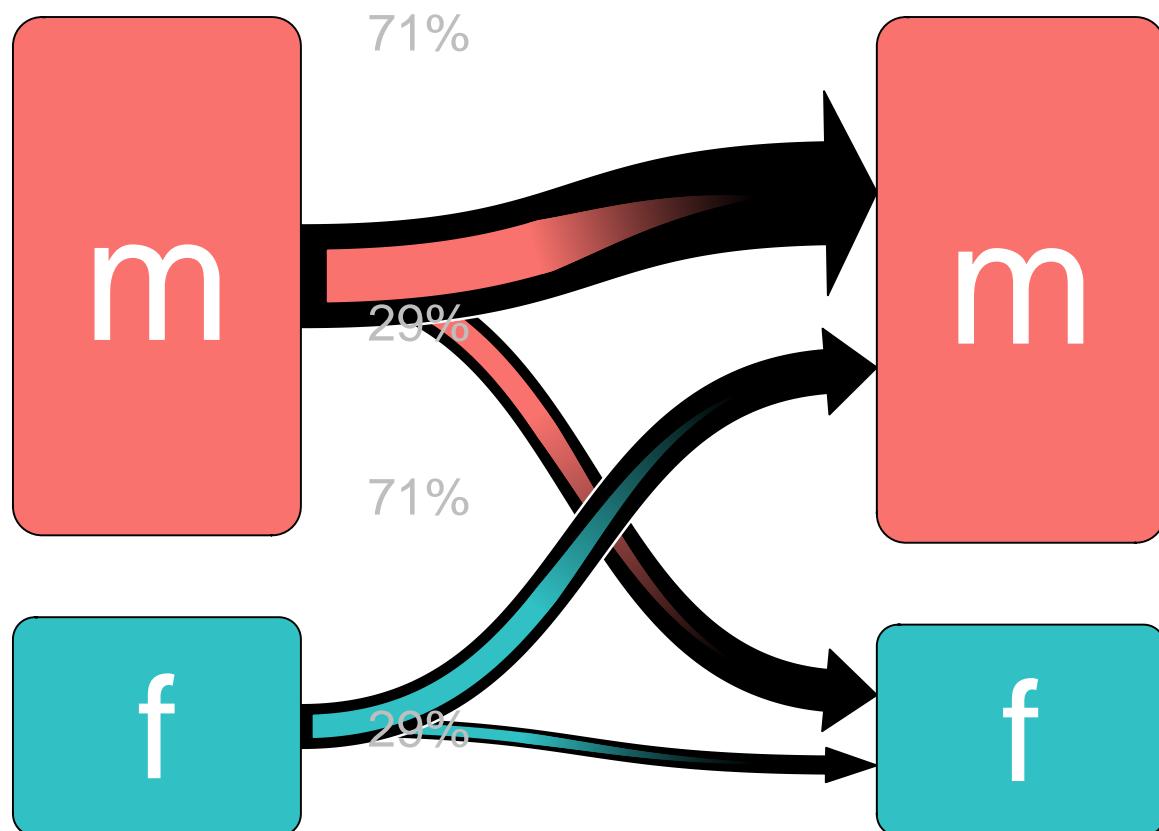
Next

Speaker



Previous Speaker Star Wars Knights of the Old Republic
Expected

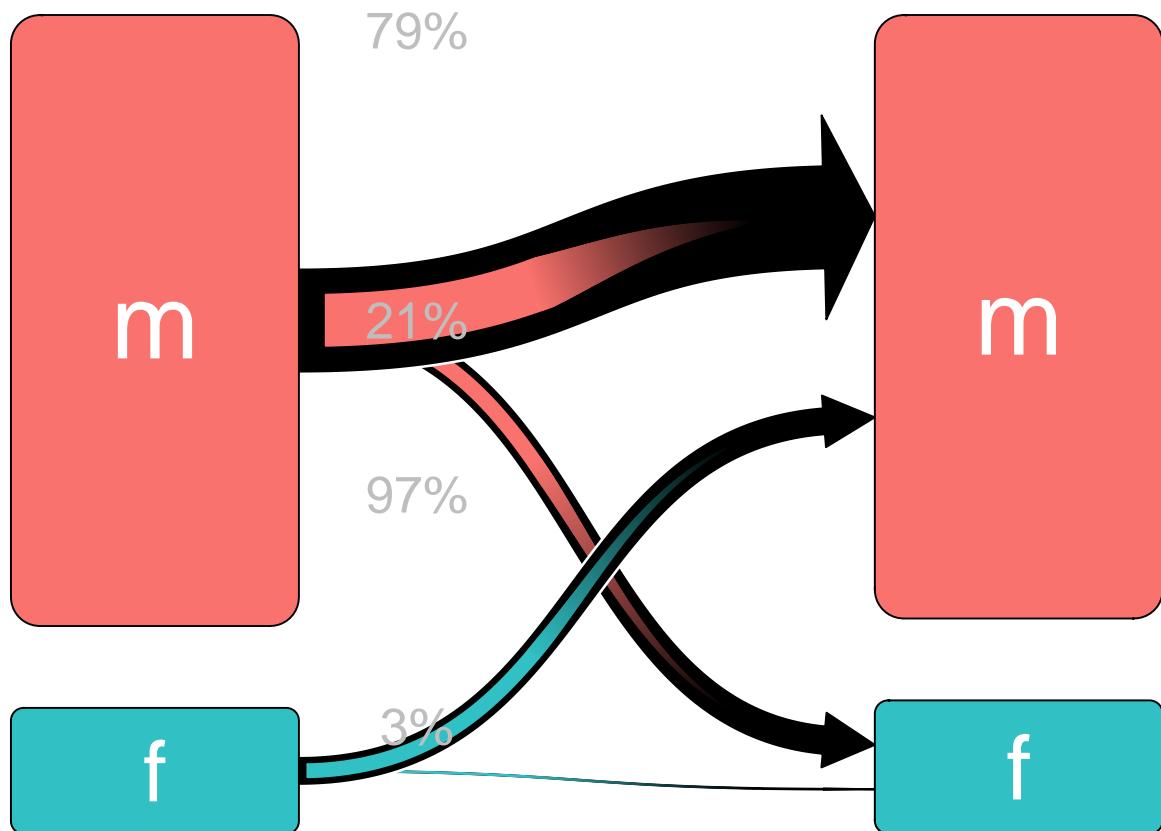
Next Speaker



Previous Speaker

Next Speaker

Super Mario RPG Legend of the Seven Stars
Empirical

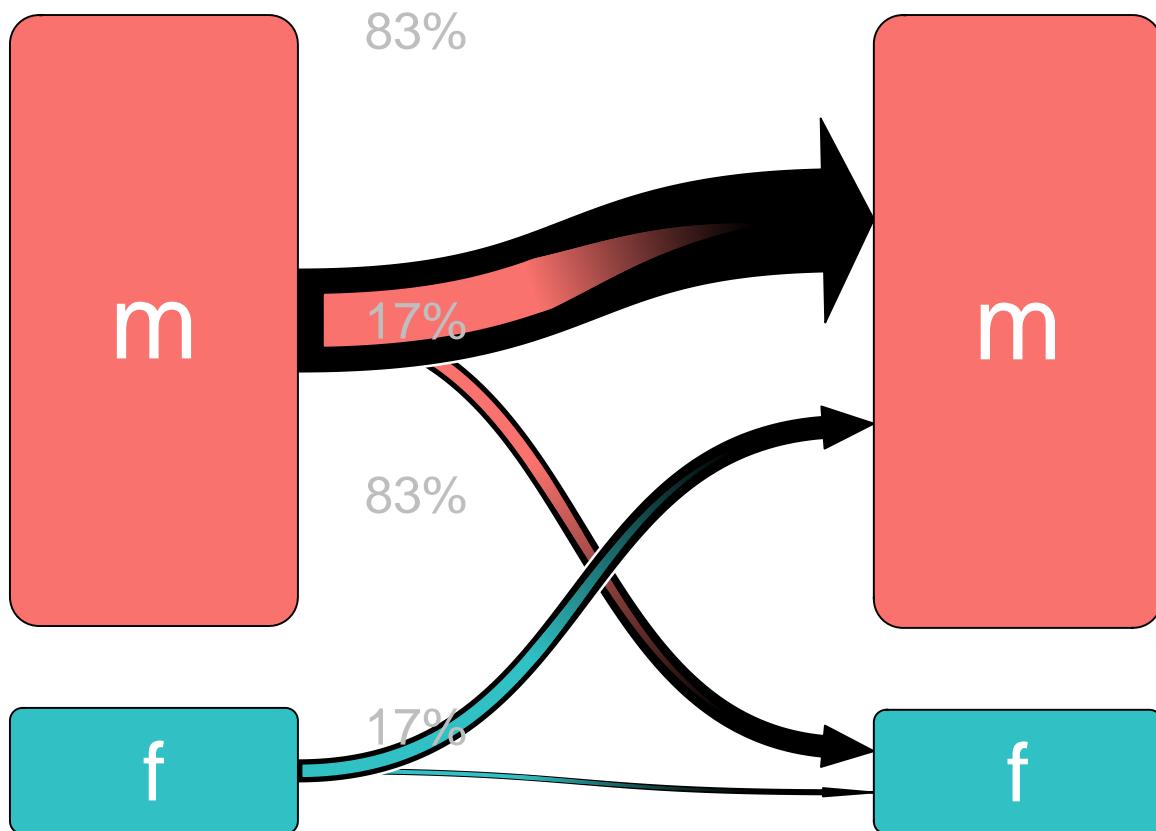


Previous Speaker

Super Mario RPG Legend of the Seven Stars

Expected

Next Speaker

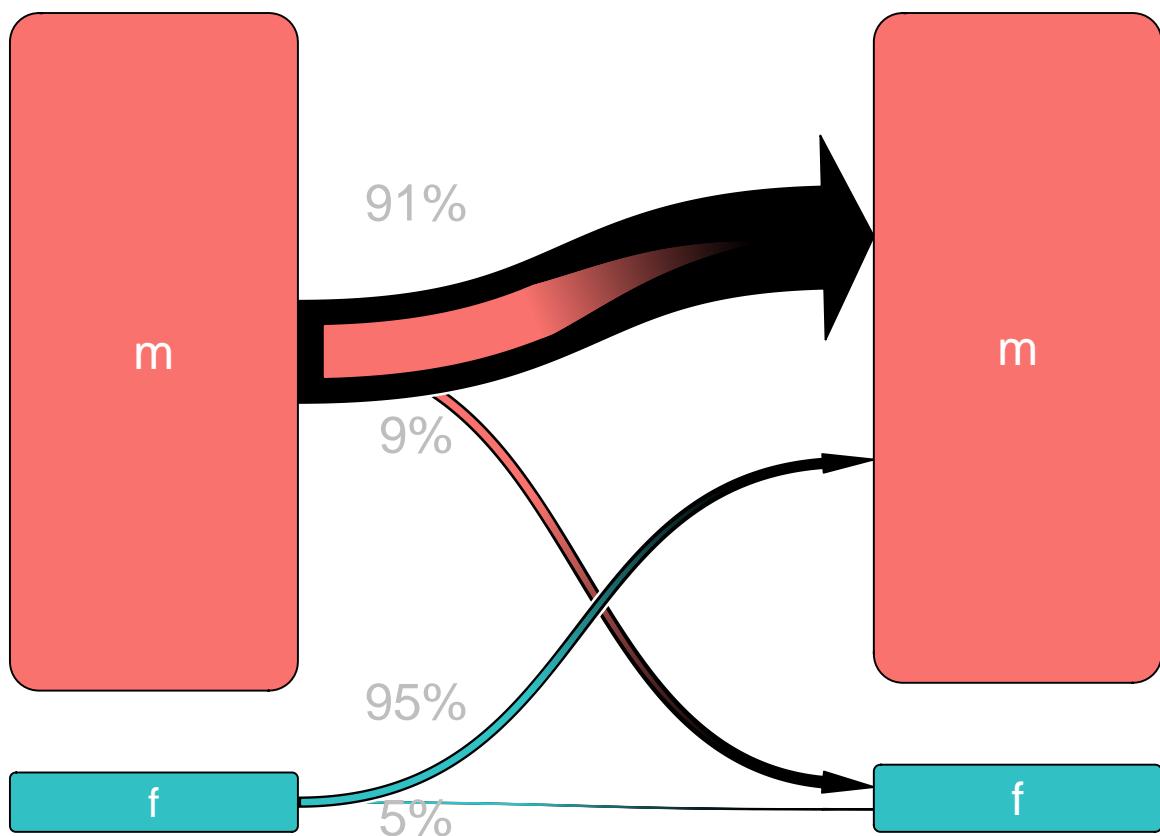


```
makeTransitionGraphForOneGame("Final Fantasy XV", ".../results/graphs/transitions/Transitions_FF15.pdf")
```

Previous
Speaker

Next
Speaker

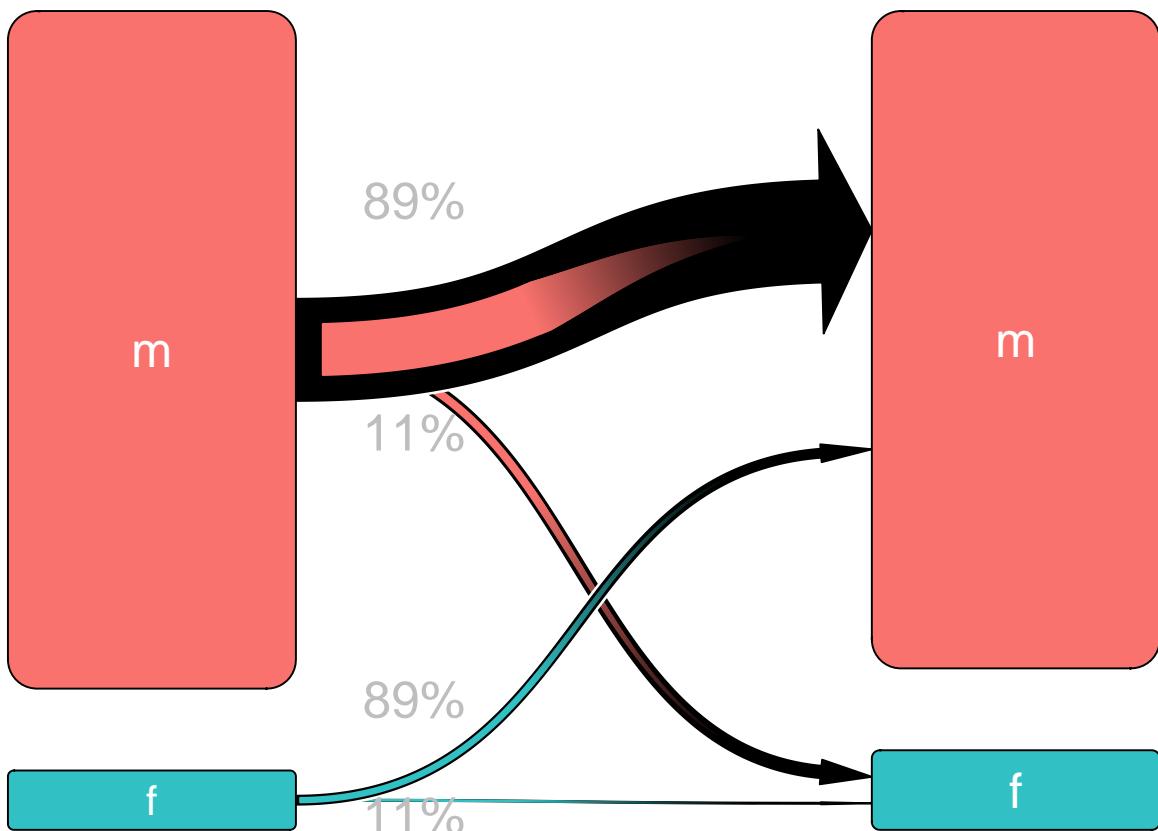
Empirical



Previous Speaker

Next Speaker

Expected



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

Write stats for paper:

```

numGames = length(unique(permuationResults$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(permuteResults$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(permuteResults$f2f.p>0.05)
cat(FFNumGamesNotBiased,
    file = "../results/latexStats/transitions_FFFNumGamesNotBiased.tex")

FFp = paste("=",permuteResults[permuteResults$game=="Final Fantasy X-2",]$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("=",permuteResults[permuteResults$game=="Final Fantasy X-2",]$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.