DAT530
Discrete Simulation and Performance Analysis
Final Project
Solitaire game strategy

Racin W. Nygaard

Universitetet i Stavanger

Abstract. SKRIV DETTE TIL SLUTT!

1 Introduction

This project aims to study the popular card game, Solitaire[Site]. Solitaire is bundled with most Windows[Site] installations, as well as being available for free on several sources. It is also easy to play the game with a physical card deck. A detailed explaination of the games rules can be found in the next chapter, Solitaire Rules[REF]
Since the game utilizes all 52 cards of the deck, the number of possible initial game states is 52!, which is a very high number. A large number of these inital game states can be merged, as they offer no difference in the difficulty to solve. Some of these intial states are unsolvable, but even given a solvable game state, one often find oneself in an unsolvable game state, due to certain actions in the game are non-reversible,. There has been attempts to find the distribution of solvable and unsolvable initial game states [ref]. This is roughly 75 percent are solvable, however the study also shows that only 35 percent of the games are won by an experienced player.
This project contains a complete model of the game, a GUI to play the game, and a basic bot to simulate user actions.

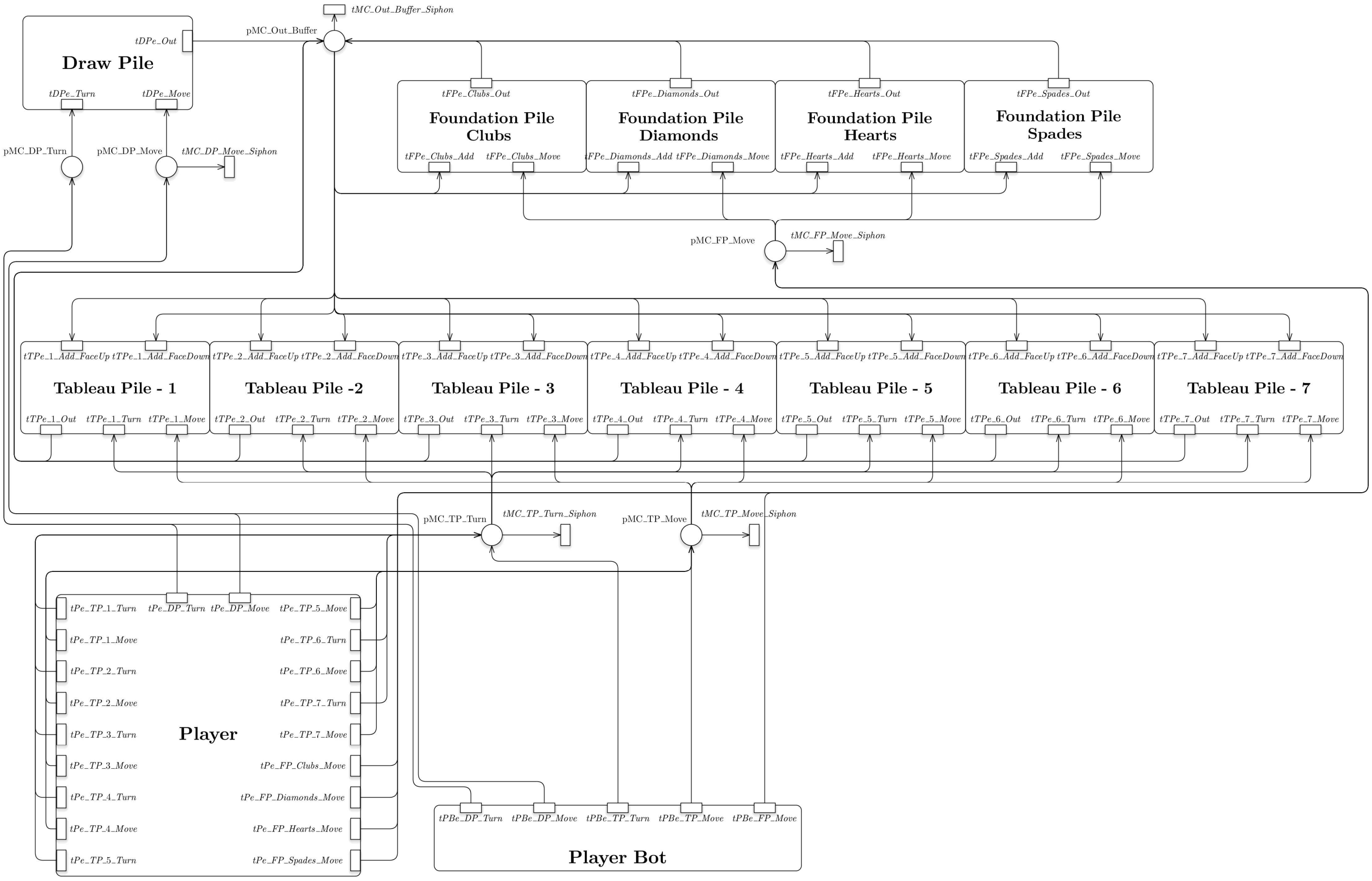
1.1 Solitaire Rules

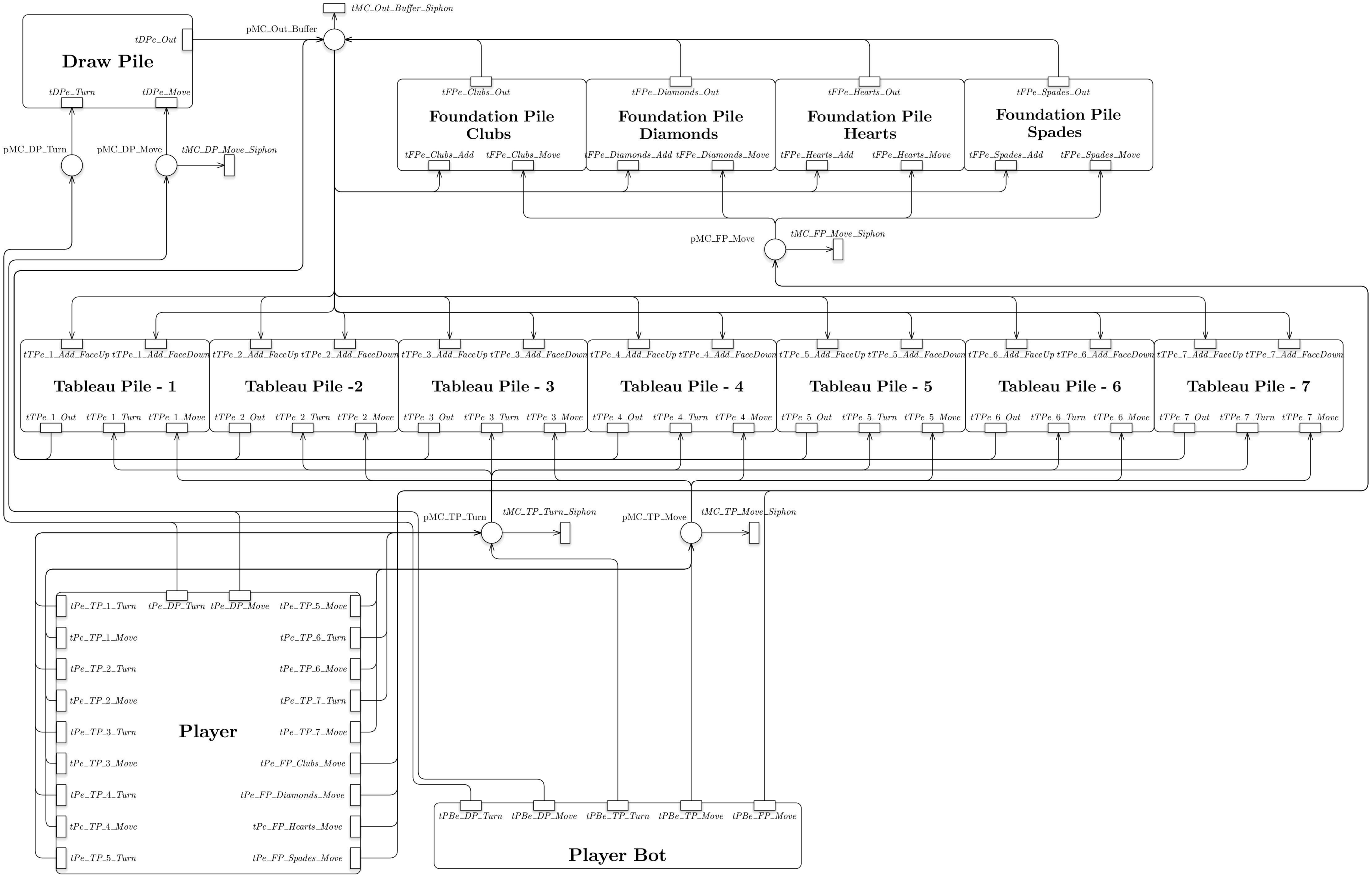
Finite State Machine ?

2 Method and Design

2.1 Naming Policy

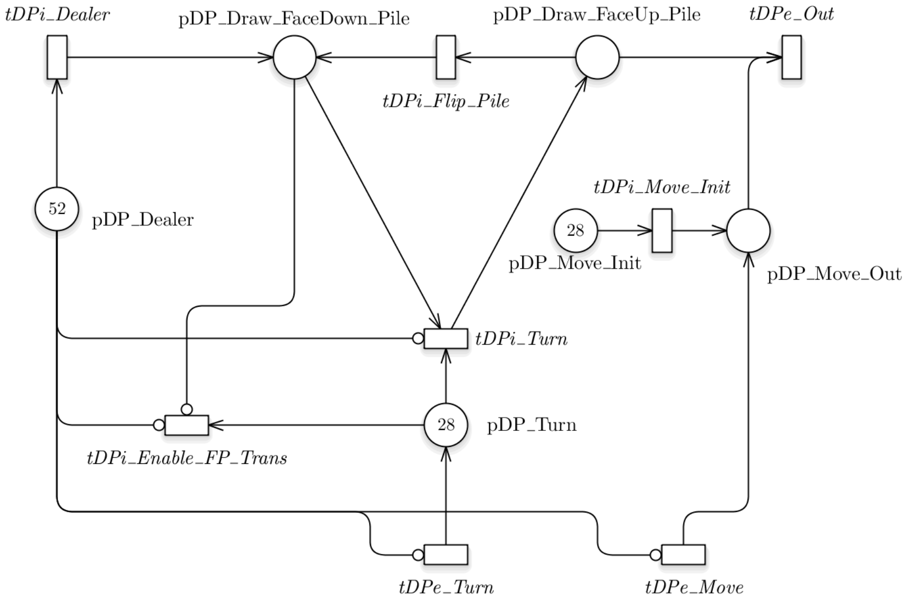
2.2 Overall Design




Draw Pile
tDPe_Out
tDPe_Turn
tDPe_Move
pMC_DP_Turn
pMC_DP_Move
tMC_DP_Move_Siphon
pMC_Out_Buffer
tMC_Out_Buffer_Siphon
Foundation Pile Clubs
tFPe_Clubs_Out
tFPe_Clubs_Add
tFPe_Clubs_Move
Foundation Pile Diamonds
tFPe_Diamonds_Out
tFPe_Diamonds_Add
tFPe_Diamonds_Move
Foundation Pile Hearts
tFPe_Hearts_Out
tFPe_Hearts_Add
tFPe_Hearts_Move
Foundation Pile Spades
tFPe_Spades_Out
tFPe_Spades_Add
tFPe_Spades_Move
pMC_FP_Move
tMC_FP_Move_Siphon
Tableau Pile - 1
tTPe_1_Add_FaceUp
tTPe_1_Add_FaceDown
tTPe_1_Out
tTPe_1_Turn
tTPe_1_Move
Tableau Pile -2
tTPe_2_Add_FaceUp
tTPe_2_Add_FaceDown
tTPe_2_Out
tTPe_2_Turn
tTPe_2_Move
Tableau Pile - 3
tTPe_3_Add_FaceUp
tTPe_3_Add_FaceDown
tTPe_3_Out
tTPe_3_Turn
tTPe_3_Move
Tableau Pile - 4
tTPe_4_Add_FaceUp
tTPe_4_Add_FaceDown
tTPe_4_Out
tTPe_4_Turn
tTPe_4_Move
Tableau Pile - 5
tTPe_5_Add_FaceUp
tTPe_5_Add_FaceDown
tTPe_5_Out
tTPe_5_Turn
tTPe_5_Move
Tableau Pile - 6
tTPe_6_Add_FaceUp
tTPe_6_Add_FaceDown
tTPe_6_Out
tTPe_6_Turn
tTPe_6_Move
Tableau Pile - 7
tTPe_7_Add_FaceUp
tTPe_7_Add_FaceDown
tTPe_7_Out
tTPe_7_Turn
tTPe_7_Move
pMC_TP_Turn
tMC_TP_Turn_Siphon
pMC_TP_Move
tMC_TP_Move_Siphon
Player
tPe_TP_1_Turn
tPe_DP_Turn
tPe_DP_Move
tPe_TP_5_Move
tPe_TP_1_Move
tPe_TP_6_Turn
tPe_TP_2_Turn
tPe_TP_6_Move
tPe_TP_2_Move
tPe_TP_7_Turn
tPe_TP_3_Turn
tPe_TP_7_Move
tPe_TP_3_Move
tPe_FP_Clubs_Move
tPe_TP_4_Turn
tPe_FP_Diamonds_Move
tPe_TP_4_Move
tPe_FP_Hearts_Move
tPe_TP_5_Turn
tPe_FP_Spades_Move
Player Bot
tPBe_DP_Turn
tPBe_DP_Move
tPBe_TP_Turn
tPBe_TP_Move
tPBe_FP_Move

The model developed is pretty large, and contains 94 transition and 42 places. It is developed using the modualar approach, and encompasses 6 different modules. Some of the modules are duplicated, with the only difference being the names of the transitions and places.

2.3 Draw Pile Module



The Draw Pile module has several key responsibilities. When first running the model, it will use the initial tokens of `pDP_Dealer` and give each of them a color according to the variable `global_info.DECK`. If the `global_info.RANDOM_DECK` is set, the cards are randomly dealt.

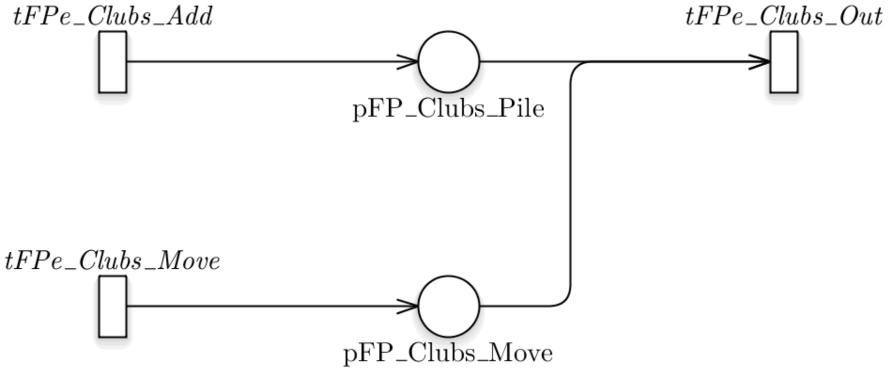
Table 1. Transitions used in Draw Pile

	Name	Description
1	tDPe_Move	
2	tDPe_Out	
3	tDPe_Turn	
4	tDPi_Dealer	
5	tDPi_Enable_FP_Trans	
6	tDPi_Flip_Pile	
7	tDPi_Move_Init	
8	tDPi_Turn	

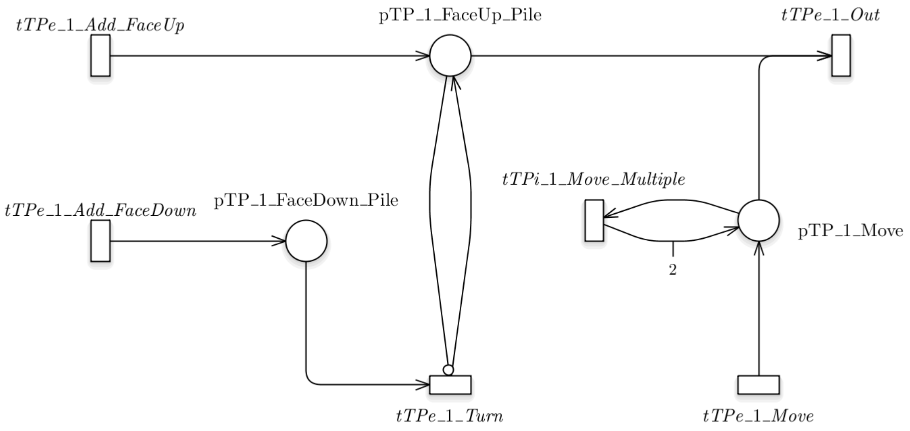
Table 2. Places used in Draw Pile

	Name	Description
1	pDP_Dealer	
2	pDP_Draw_FaceDown_Pile	
3	pDP_Draw_FaceUp_Pile	
4	pDP_Move_Init	
5	pDP_Move_Out	
6	pDP_Turn	

2.4 Foundation Pile Module



2.5 Tableau Pile Module



2.6 Module Connector Module

2.7 Player Module

2.8 Player Bot Module

3 Implementation

3.1 Algorithms

Atomicity In order to preventdd

3.2 Initial Dealing

3.3 Resources

3.4 Moving Multiple Cards

```
def mapper_from_to(self, key, email):
    if 'to' in email.keys() and 'from' in email.keys() and 'body_count' in email.key
```

4 Testing, Analysis and Results

4.1 Algorithms

Atomicity In order to preventdd

4.2 Initial Dealing

4.3 Resources

4.4 Moving Multiple Cards

5 Discussion

References

1. Wikipedia article on Tf-idf. https://en.wikipedia.org/wiki/Tf?idf
2. Tom White, Hadoop: The Definitive Guide, 2015, *ISBN: 978-1-491-90163-2*
3. Docker API Docs, https://docs.docker.com
4. Slides from DAT630, Krisztian Balog
5. Kaggle. The Enron Email Dataset. https://www.kaggle.com/wcukierski/enron-email-dataset
6. Data Intensive Systems Compendium, Tomasz Wiktorski et al.
7. Source code of all tasks developed. GitLab https://gitlab.com/mindejulian/projectDAT500/tree/master