Caroline Sayuri Kagohara

SIMULAÇÃO E COMPARAÇÃO DE POLÍTICAS DE TOMADA DE DECISÃO PARA O SISTEMA DE ELEVADORES DO INSTITUTO DE CÂNCER DO ESTADO DE SÃO PAULO UTILIZANDO O SOFTWARE ANYLOGIC

São Paulo

2020

CAROLINE SAYURI KAGOHARA

SIMULAÇÃO E COMPARAÇÃO DE POLÍTICAS DE TOMADA DE DECISÃO PARA O SISTEMA DE ELEVADORES DO INSTITUTO DE CÂNCER DO ESTADO DE SÃO PAULO UTILIZANDO O SOFTWARE ANYLOGIC

Trabalho de Formatura apresentado à Escola Politécnica da Universidade de São Paulo para a obtenção do diploma de Engenheira de Produção

Orientador: Prof. Dr. Daniel de Oliveira Mota

São Paulo

2020

Catalogação-na-publicação

KAGOHARA, C.
SIMULAÇÃO E COMPARAÇÃO DE POLÍTICAS DE TOMADA DE DECISÃO PARA O SISTEMA DE ELEVADORES DO INSTITUTO DE CÂNCER DO ESTADO DE SÃO PAULO UTILIZANDO O SOFTWARE ANYLOGIC / C. S. K.
-- São Paulo, 2020
102 p.

Trabalho de Formatura - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Produção.

1.Simulação de elevadores I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Produção II.t.

À minha família.

AGRADECIMENTOS

Primeiramente, gostaria de agradecer aos meus pais, Miriam e Humberto, que me deram todas as oportunidades e condições para chegar até aqui. Serei eternamente grata.

Também gostaria de agradecer aos meus irmãos, Juliane e Nicholas, que sempre me apoiaram e fizeram parte da construção da pessoa que sou hoje. E à minha avó, Selma, por sempre me incentivar a estudar e a ir atrás dos meus sonhos.

Um agradecimento especial ao Lucas, namorado e companheiro, que esteve sempre presente nesses últimos anos, apoiando minhas decisões, me consolando e comemorando comigo e me ensinando muito sobre como ser uma pessoa e uma engenheira melhor.

Aos meus amigos, ou como gosto de chamar "melhores presentes que a Poli me deu", Clari, Bruno, Miky e Karen, por estarem sempre presentes nesses anos de faculdade, trabalhos em grupo, estudos, conversas do dia-a-dia e até reflexões de vida. Muito obrigada por vivenciarem tudo isso comigo.

Aos queridos Bruna, Natalia, Gustavo e Victoria, que desde o ensino médio torcem pelo meu sucesso, me apoiam e me acompanham, mesmo que não tão perto quanto antes, mas com o mesmo carinho de sempre. Obrigada por todos esses anos de amizade!

Também gostaria de agradecer ao time de tênis de mesa da Poli, com certeza foram essenciais nesses anos e me ensinaram muito sobre perseverança, objetivos e trabalho em equipe. Obrigada por me permitirem continuar praticando um esporte que mudou a minha vida e por me proporcionarem momentos emocionantes e inesquecíveis. E um obrigada especial ao César e ao Luiz, por estarem comigo não só nos jogos, mas nas "competições" da vida.

E por último, mas de extrema importância, ao meu orientador, Daniel Mota. Obrigada por todo o seu entusiasmo com o projeto. Essa empolgação com certeza faz toda a diferença e traz motivações aos seus alunos e à sua equipe de orientados.

“A persistência é o menor caminho do êxito”

Charles Chaplin

RESUMO

Todos os dias, milhares de pessoas transitam pelo Instituto de Câncer do Estado de São Paulo e tratando-se de um prédio com 27 andares, a maioria dessas pessoas utilizam os elevadores do prédio para sua locomoção por entre os pavimentos. Pensando nisso, o presente estudo busca montar um modelo de simulação para o sistema de elevadores centrais do prédio, composto por 6 cabines, utilizando as especificações de funcionamento dos elevadores e as informações sobre o trânsito de pessoas pelo prédio, fornecidos pela equipe de segurança do instituto.

Um sistema de elevadores pode atuar com diferentes regras, ou políticas de tomada de decisão, que determinam a sequência de chamados a qual os elevadores devem seguir. Este trabalho busca comparar três políticas de tomada de decisão sob o ponto de vista de quatro indicadores principais: tempo de espera pela chegada dos elevadores, tempo de viagem, taxa de ocupação dos elevadores e tamanho da fila de espera em cada andar do prédio.

Para tal estudo foi utilizado o software AnyLogic, muito utilizado para a realização de simulações de diferentes processos em diferentes tipos de indústria, com o auxílio das ferramentas Excel e Python para a análise dos dados.

Palavras-chave: sistema de elevadores, políticas de tomada de decisão, AnyLogic, simulação

ABSTRACT

Every day, thousands of people pass through the Instituto de Câncer do Estado de São Pualo and dealing with a 27 floors building, most of these people use the elevators to move around the floors. Having that in mind, this study seeks to assemble a simulation model for the building's central elevators system, consisting of 6 cabins, using the elevator's operating specifications and information about the transit of people through the building as provided by the security team of the institute.

An elevator system can operate with different rules, or decision-making policies, that determine the sequence of calls that elevators must follow. This paper seeks to compare three decision-making policies from the point of view of four main indicators: waiting time for the arrival of the elevators, travel time, occupancy rate of the elevators and the size of the queue on each floor of the building.

For this research, the AnyLogic software was used, widely used to carry out simulations of different processes in different types of industry, with the aid of Excel and Python tools for data analysis.

Keywords: elevators system, decision-making policies, AnyLogic, simulation

LISTA DE FIGURAS

Figura 1 – Planta de um dos andares do prédio com a localização dos elevadores de estudo em destaque 27
Figura 2 – Funções de associação para as características de tráfego de passageiros (retirado de *ChangBum Kim et al, 1998*) 39
Figura 3 – Representação dos termos do domínio da adequação (retirado de *ChangBum Kim et al, 1998*) 41
Figura 4 – Utilizando o Sistema ETA 41
Figura 5 – Utilizando Fuzzy Logic 42
Figura 6 – Utilizando o Sistema ETD 43
Figura 7 – Nível de abstração de exemplos de modelos de simulação 48
Figura 8 – Paradigmas mais adequados para diferentes níveis de abstração 48
Figura 9 – Tabela de acionamentos dos elevadores do ICESP no dia 05 de outubro (fonte: fornecido pela equipe de segurança do ICESP) 53
Figura 10 – *Software* AnyLogic (retirado no site do AnyLogic) 56
Figura 11 – Fluxo de processos do elevador 57
Figura 12 – Fluxo de processos dos elevadores simplificado 57
Figura 13 – Exemplo de função em Java para complemento das funções de um determinado bloco 58
Figura 14 – Blocos interligados representando o fluxo de processos pelos quais os elevadores passam 58
Figura 15 – Argumentos que caracterizam os elevadores 59
Figura 16 – Argumentos que caracterizam os passageiros 60
Figura 17 – Representação dos andares do prédio (andares: 17, 18, 19, 20, 21 e 22) 62
Figura 18 – Componentes .txt para armazenamento dos dados resultados da simulação 74
Figura 19 – Histograma do tempo de espera na fila – política 1 76
Figura 20 - Histograma do tempo de permanência no sistema – política 1 77
Figura 21 – Histograma do tempo de espera na fila – política 2 77
Figura 22 – Histograma do tempo de permanência no sistema – política 2 78
Figura 23 – Histograma do tempo de espera na fila – política 3 78
Figura 24 – Histograma do tempo de permanência no sistema – política 3 79

LISTA DE TABELAS

Tabela 1 – Classificação do tráfego de passageiros (adaptado de *ChangBum Kim et al, 1998*) 38
Tabela 2 – Características do tráfego de passageiros (adaptado de *ChangBum Kim et al, 1998*) 39
Tabela 3 – Classificação dos tipos de pesquisa (adaptado do *professor Robson Bonomo*).....50
Tabela 4 – Dados constantes do ICESP (fonte: conversa com equipe de segurança do ICESP) 54
Tabela 5 – Probabilidade acumulada dos andares de chegada dos passageiros 65
Tabela 6 – MAPE médio para cada uma das três políticas de decisão.....71
Tabela 7 – Probabilidade de cada andar ser o andar de entrada de um passageiro 72
Tabela 8 – Resultados da simulação – tempo de espera na fila e tempo de permanência no sistema - para cada política de decisão.....76
Tabela 9 – Taxa de ocupação do elevador para cada política de tomada de decisão 79
Tabela 10 – Tamanho da fila de espera para cada andar e para cada política de tomada de decisão 80
Tabela 11 – Resultados das simulações considerando diferentes velocidades de movimentação dos elevadores 81

LISTA DE SIGLAS E ABREVIAÇÕES

ICESP – Instituto de Câncer do Estado de São Paulo

EGCS – Sistema de Controle de Grupo de Elevadores (Elevator Group Control System)

ETA – Estimated Time of Arrival

ETD – Estimated Time to Destination

SDF – System Degradation Factor

UPT – Up Traffic

DNT – Down Traffic

CITP – Centralized In Traffic Percentage

DOTP – Distributed Out Traffic Percentage

BT – Business Time

UP – Up Peak

DP – Down Peak

LT-A – Lunch Time A

IT – Inactive Time

LT-B – Lunch Time B

BTH – Business Time and Heavy Traffic

HT – Heavy Traffic

AWT – Avarage Waiting Time

LWP – Long Waiting Percentage

RNC – Run Cost

HCWT – Hall Call Waiting Time

MaxHCWT – Maximum Hall Call Waiting Time

CV – Coverability

GD – Gathering Degree

SUMÁRIO

1. INTRODUÇÃO 25

1.1. Problema Estudado 25

1.2. Objetivo do Trabalho 26

1.3. Justificativa 26

1.3.1. Local e Área de Estudo 26

1.4. Estrutura 28

2. REVISÃO BIBLIOGRÁFICA 29

2.1. Sistema de Elevadores 29

2.1.1. Elevadores de uma cabine 29

2.1.2. Sistema de Controle de Grupo de Elevadores 32

2.1.2.1. Estimated Time of Arrival (ETA) 33

2.1.2.2. Estimated Time to Destination (Destination Dispatching) 35

2.1.2.3. Fuzzy Logic 36

2.1.2.4. Comparação entre os sistemas 41

2.1.2.5. Estratégias de Controle 44

2.2. Modelagem Matemática 44

2.2.1. Principais Paradigmas da Simulação 47

2.2.1.1. Sistemas Dinâmicos 48

2.2.1.2. Eventos Discretos 49

2.2.1.3. Modelos Baseados em Agentes 49

3. METODOLOGIA 50

3.1. Metodologia de Pesquisa 50

3.1.1. Apresentação da Metodologia Escolhida 50

3.2. Metodologia do Projeto 52

3.2.1. Fonte de Dados 52

3.2.1.1. Descrição das Variáveis de Estudo 52

3.2.1.2. Restrições dos Dados 54

3.2.2. Ferramentas de Estudo Utilizadas 54

3.2.2.1. AnyLogic 55

4. PROJETO 57

4.1. Elaboração do modelo de simulação 57

4.1.1. Estrutura do modelo 58

4.1.2. Premissas consideradas e funções complementares em cada bloco 63

4.1.3. Políticas de decisão 69

4.1.5. Validação do modelo 70

4.2. Tratamento Inicial dos Dados de Entrada 71

5. RESULTADOS 74

5.1. Coleta de dados da simulação 74

5.2. Tratamento dos Dados de Saída 75

5.3. Resultados da simulação 75

6. CONCLUSÃO 82

REFERÊNCIAS BIBLIOGRÁFICAS 83

ANEXO 1 – Código da função que determina o andar de saída dos passageiros 86

ANEXO 2 – Tabela de probabilidades para cada andar de saída 98

1. INTRODUÇÃO

Nos últimos anos, a curva de crescimento populacional do Brasil se mantém ascendente, assim como a população do Estado de São Paulo. Segundo dados do IBGE (consulta no dia 07 de maio de 2020), 46.229.081 pessoas vivem no Estado e esse número tende a crescer, chegando a mais de 49 milhões de pessoas em 2030. Junto a esse crescimento, vem a necessidade da construção de prédios cada vez mais altos para suprir a necessidade de comportar essa população, o que é visivelmente percebido nas ruas de São Paulo. A cidade é considerada uma das 10 maiores metrópoles do mundo com quase 6 mil prédios, dos quais 12 possuem mais de 150 metros de altura.

Esse crescimento no número de grandes edifícios e arranha-céus é acompanhado pela necessidade de mobilidade vertical, fazendo com que as pessoas tenham que se adaptar para chegar a distâncias cada vez mais altas, isso é refletido no aumento da demanda por elevadores. E ainda, juntamente ao grande dinamismo das metrópoles, exige que esses elevadores sejam cada vez mais rápidos e eficientes.

Essa mesma evolução pode ser observada em hospitais. O que, até a década de 20, era organizado horizontalmente, dividido em pavilhões diversos para separar os diferentes tipos de enfermidades e evitar a transmissão de doenças entre os pacientes, passou a ser verticalizado, com o isolamento feito em diferentes pavimentos. E analogamente ao dinamismo das metrópoles que exige a eficiência e rapidez dos elevadores, as emergências encontradas em hospitais e pronto socorro exigem o mesmo, tanto para a locomoção de profissionais, pacientes e macas, como para a entrega de equipamentos e medicamentos.

Tendo isso em mente, muitos hospitais vêm investindo em sistemas de elevadores inteligentes, que se mostraram mais eficientes se controlados por um sistema único cujo objetivo é melhorar a distribuição de elevadores entre as chamadas realizadas e atendê-las mais rápido. Um hospital que adotou este sistema foi o ICESP (Instituto de Câncer do Estado de São Paulo), que será o objeto de estudo do presente trabalho.

1.1. Problema Estudado

Na primeira visita feita ao ICESP, percebeu-se que o tempo de espera pela chegada do elevador ao andar solicitado é excessivo. Além disso, os próprios frequentadores do instituto comentam que essa demora é um problema.

Em outra reunião feita com a equipe responsável pelo controle dos elevadores, foram relatadas algumas das tentativas para a redução do tempo de espera, assim como seus respectivos resultados. Estes considerados insatisfatórios pela equipe, uma vez que se trata de

um instituto com renome no tratamento de câncer, no qual circulam milhares de pessoas por dia.

Dessa maneira, o problema a ser estudado no presente trabalho é o tempo de espera pela chegada dos elevadores do ICESP ao andar solicitado.

1.2. Objetivo do Trabalho

Com o intuito de resolver o problema estudado, o objetivo do trabalho é entender o funcionamento do sistema de elevadores do ICESP, dimensionar o sistema em um modelo matemático do fluxo de pessoas pelos 27 andares do prédio e comparar diferentes políticas de tomada de decisão dos elevadores.

1.3. Justificativa

O mundo atual está cada vez mais dinâmico, as pessoas têm pressa para chegar aos seus destinos e a necessidade por mobilidade cresce simultaneamente. A verticalização dos edifícios requer que essa mobilidade também se verticalize e seja tão eficiente quanto.

Em hospitais essa necessidade é ainda mais acentuada, pois além da correria popular habitual, existem muitos casos de emergências de pacientes passando mal e que precisam de atendimento imediato. No ICESP não é diferente, passam por lá milhares de pessoas por dia e muitas delas estão em condições graves.

Dessa forma, o estudo do sistema de elevadores do ICESP é fundamental para o bom fluxo de funcionários e pacientes por entre os andares do prédio, buscando garantir a maior eficiência dos elevadores possível.

1.3.1. Local e Área de Estudo

O trabalho foi realizado no Instituto de Câncer do Estado de São Paulo (ICESP), localizado na região central da capital, e que compõe o complexo do Hospital das Clínicas da Faculdade de Medicina da Universidade de São Paulo, reconhecido como o mais importante centro de atendimento médico da América Latina.

O ICESP é um instituto voltado exclusivamente para casos oncológicos e pacientes com diagnóstico de câncer confirmado. Teve suas atividades iniciadas em 2008 como uma Organização Social de Saúde e possui como missão “Ser um centro de excelência, promovendo o ensino, a pesquisa e a assistência médico hospitalar na área do câncer, de acordo com os princípios definidos pelo Sistema Único de Saúde (SUS), visando contribuir com a saúde e a qualidade de vida da sociedade”, (retirado do site oficial do ICESP).

O edifício tem 112 metros de altura, sendo um dos maiores hospitais verticais do mundo, com 28 andares que abrangem diferentes tipos de serviços médicos voltados a casos de câncer.

Para atender a estes andares, o instituto conta com 17 elevadores, sendo 3 para a Torre Laranja, 3 para a Torre Azul, 1 para o transporte de alimentos, 1 para o transporte de medicamentos, 1 para o transporte de roupas sujas, 1 para o transporte de lixo, 1 que atende apenas o último andar e o heliporto e 6 elevadores na torre central para o transporte de funcionários e pacientes.

Os elevadores da torre central são controlados por um sistema de controle único inteligente e serão os objetos de estudo do trabalho.

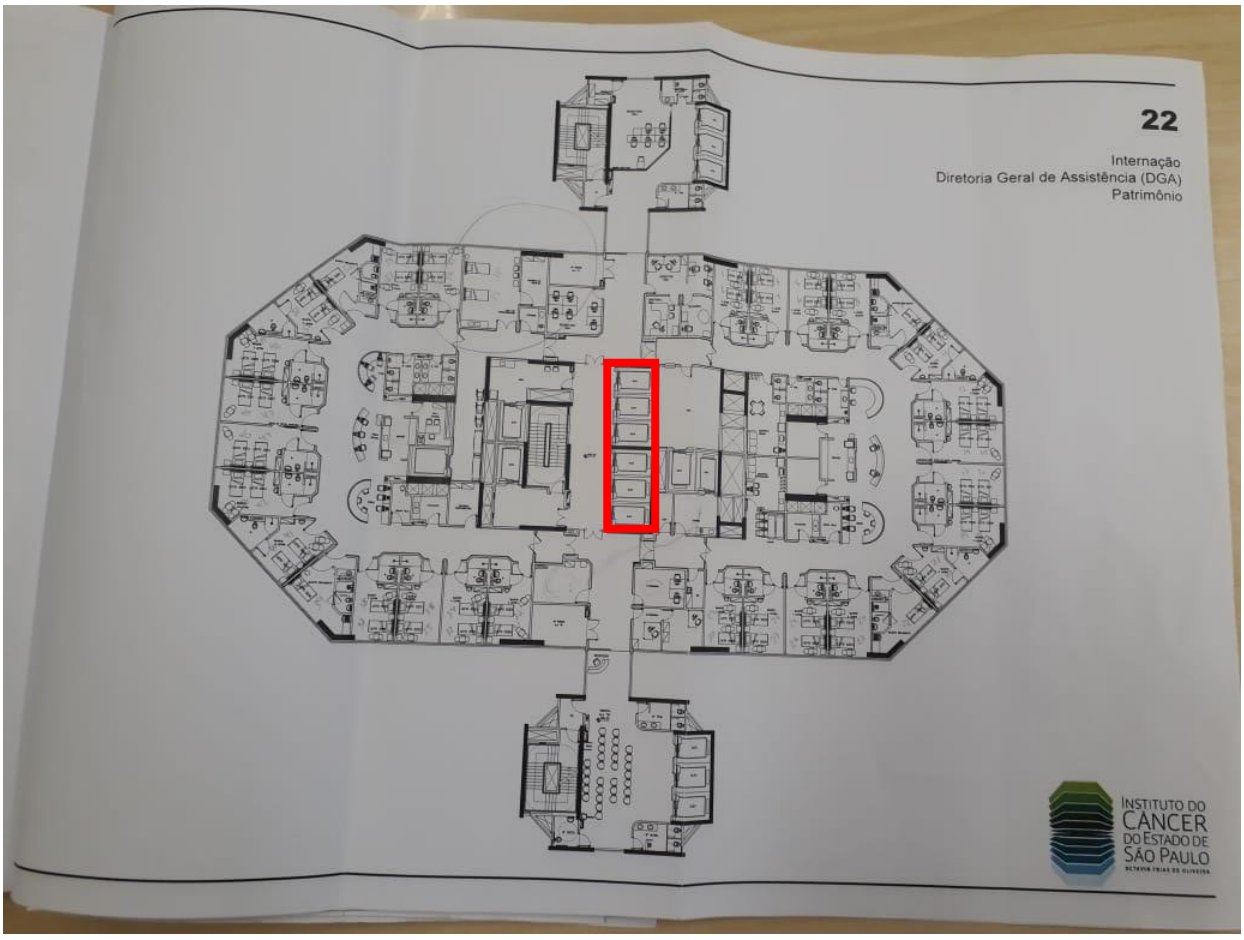


Figura 1 – Planta de um dos andares do prédio com a localização dos elevadores de estudo em destaque

Além disso, o instituto conta com um correio pneumático composto por uma tubulação preparada utilizando ar comprimido para o transporte de amostras sanguíneas, medicamento de alto custo, documentos ou pequenos materiais.

1.4. Estrutura

O presente trabalho está dividido em seis capítulos: 1. Introdução, 2. Revisão bibliográfica, 3. Metodologia, 4. Projeto, 5. Resultados e 6. Conclusão.

A introdução apresenta a temática do estudo de um sistema de elevadores e sua contextualização no Instituto de Câncer do Estado de São Paulo.

Na revisão bibliográfica são abordados dois pontos principais: Sistema de Elevadores, começando pelo entendimento do funcionamento de um único elevador e algumas premissas adotadas e ampliando o assunto para um sistema de elevadores e alguns métodos de controle, e Modelagem Matemática, onde são apresentados o conceito de modelagem sob a visão de diferentes estudiosos e os principais paradigmas de uma simulação.

Em metodologia, são descritas as técnicas e métodos utilizados na pesquisa e também apresentada a ferramenta de simulação.

No capítulo de projeto, o modelo de simulação é detalhado e validado e são apresentadas as premissas consideradas na simulação, assim como os dados de entrada e variáveis de estudo e as políticas de tomada de decisão.

Em resultados, são mostrados os indicadores resultantes da simulação e a comparação das três políticas escolhidas.

E por fim, na conclusão, analisa-se qualitativamente o trabalho e são feitos comentários sobre os resultados e limitações encontradas.

2. REVISÃO BIBLIOGRÁFICA

Para o bom entendimento do sistema de elevadores do ICESP e a modelagem matemática do fluxo de pessoas no instituto foi feito o levantamento de algumas referências empregadas por autores que estudaram temas relacionados ao assunto e/ou trabalharam com a simulação de sistemas de elevadores em outros edifícios. Essas referências foram separadas em três temas: 1. Sistema de Elevadores: onde foi compreendido, primeiramente, o funcionamento de um elevador individual, seus princípios de atendimento e parâmetros que devem ser levados em consideração no seu estudo e, em seguida, foi entendido o conceito de um sistema de elevadores e seu processo de tomada de decisões; 2. Fluxo de pessoas em hospitais: para a compreensão do trânsito de pessoas entre os andares do instituto e entendimento da demanda pelo uso dos elevadores; 3.Modelagem Matemática: onde foram apresentados conceitos de simulação através de software e foram levantadas as referências de pesquisadores que já realizaram a modelagem de um sistema de elevadores.

2.1. Sistema de Elevadores

2.1.1. Elevadores de uma cabine

Antes de entender o funcionamento de um sistema de elevadores, é importante compreender a performance de um elevador individualmente. Um elevador simples, composto por uma única cabine, pode encontrar-se em dois estados: em movimento ou parado em algum andar. Este elevador pode receber um chamado de algum andar (neste trabalho chamaremos de *hall call* quando algum passageiro está em um determinado andar e chama o elevador) e dirigir-se para o determinado andar ou apenas abrir as suas portas caso já se encontre no devido andar. Após a abertura das portas, o(s) passageiro(s) entra no elevador e seleciona o andar para o qual deseja ir (neste trabalho será chamado de *cabin call* quando o passageiro escolhe o andar ao qual deseja chegar) e então as portas se fecham e o elevador dirige-se ao andar escolhido.

É possível descrever a atividade de um elevador em 18 eventos: 1. Passageiro aperta o botão; 2. Sistema do elevador registra a chamada; 3. Elevador se desloca ao andar da chamada; 4. Elevador desacelera próximo ao andar da chamada; 5. Elevador para no andar da chamada; 6. Porta começa a se abrir; 7. Porta termina de abrir; 8. Passageiro entra; 9. Passageiro seleciona o andar de destino; 10. Sistema registra o andar de destino; 11. Porta começa a se fechar; 12. Porta se fecha; 13. Elevador começa a se movimentar; 14. Elevador desacelera próximo ao andar de destino; 15. Elevador para no andar de destino; 16. Porta começa a abrir; 17. Porta termina de abrir; 18. Passageiro sai do elevador.

Os elevadores de funcionamento individual mais comuns são caracterizados por possuírem um painel interno na cabine com botões de chamada para todos os andares do prédio e um botão em cada andar para a solicitação do elevador (*hall call*). E para poder responder ao chamado, os elevadores, comumente, atendem à algumas premissas:

- Os elevadores possuem uma capacidade máxima de peso ou quantidade de pessoas que conseguem transportar e não fecham suas portas caso essa capacidade tenha sido excedida;
- Os elevadores não se movimentam com suas portas abertas;
- As portas dos elevadores só são abertas nos andares em que houve alguma solicitação (podendo ser um *cabin call* ou um *hall call*);
- *Cabin calls* sempre são atendidos antes dos *hall calls*; (*Gina Barney and Lutfi A-sharif, 2016*);
- Os *cabin calls* são atendidos sequencialmente conforme o sentido atual do seu movimento (sequência crescente dos andares se o elevador estiver subindo e decrescente se estiver descendo);
- O elevador não muda o sentido do seu movimento enquanto houver passageiros embarcados. (*Gina Barney and Lutfi A-sharif, 2016*)

Tendo essas premissas em mente, o elevador deve decidir qual a sequência em que atenderá aos chamados, seguindo um determinado algoritmo de decisão com o qual foi programado. No livro *Elevator Traffic Handbook*, *Gina Barney* e *Lutfi Al-Sharif (2016)* apresentam o controle automático de chamada única, também chamado de *noncollective* ou *automatic pushbutton control*, no qual o próprio passageiro opera o elevador através dos botões no painel da cabine e dos localizados em cada pavimento do edifício e nenhum ascensorista é necessário. Se o passageiro aperta um botão do painel de controle da cabine referente ao andar de destino, o elevador se dirige a esse andar sem parar em nenhum outro pavimento no meio do trajeto, mesmo que algum outro usuário tenha realizado um outro *hall call*. E assim que o primeiro passageiro tiver desembarcado, o segundo *hall call* é atendido. Este tipo de controle é útil apenas em edifícios de pequeno porte, com pouco fluxo de passageiros.

Outro tipo de controle apresentado é o controle coletivo, uma designação genérica para o tipo de comando onde todos os chamados (tanto os *cabin calls* quanto os *hall calls*) são registrados e atendidos. O elevador para automaticamente nos pavimentos de desembarque seguindo a ordem dos andares e não a ordem na qual os botões foram pressionados. Esse tipo de controle pode ser composto por um botão de chamada por pavimento ou dois (um para

subidas e outro para descidas). Com um botão, o passageiro registra o *hall call* independentemente do atual sentido de viagem do elevador e este irá atender ao chamado. Lembrando que, se o elevador já estiver transportando um passageiro, o sentido da viagem não irá ser alterado, independentemente do destino do usuário que apertou o botão, deixando a este a escolha entre entrar no elevador mesmo que a viagem seja no sentido oposto ao seu destino, ou não entrar e esperar que o elevador fique livre ou esteja dirigindo-se no sentido desejado, entretanto pode resultar em um elevado tempo de espera. Este tipo de controle pode ser adaptado para edifícios nos quais há tráfego de pessoas apenas entre o andar térreo e os demais andares, sem viagens intermediárias, sendo adequado utilizar o controle do tipo *down collective* (ou *up-distributive*), onde é considerado que todos os *hall calls* tem como destino pisos inferiores ao de origem. Dessa forma, o elevador sobe diretamente até o andar mais alto em que houve algum *hall call* registrado e segue descendo atendendo aos demais chamados. O mesmo pode ser aplicado considerando todos os destinos em pisos superiores, com origem no piso mais inferior.

Com dois botões por andar, um de subida e outro de descida, é utilizado outro tipo de comando, o *full collective* (ou *directional colletive*), em que o passageiro deve selecionar apenas a direção para a qual deseja ir. O elevador irá parar para atender apenas aos chamados (tanto *hall call* quanto *call hall*), cujas viagens sejam no mesmo sentido que o atual realizado pela cabine. Quando não houver mais chamados neste sentido, o elevador se dirigirá ao andar mais distante que houver algum chamado na direção oposta e irá reverter seu deslocamento para atender as chamadas na nova direção. Este tipo de controle também é utilizado em sistemas *duplex* (dois elevadores), *tríplex* (três elevadores) ou *quadruplex* (quatro elevadores).

Além da sequência em que os chamados serão atendidos, diversos outros critérios devem ser considerados na caracterização de um elevador. *Marja-Liisa Siikonen (2000)* estudou o tráfego de pessoas em edifícios altos e fez o levantamento dos seguintes parâmetros em relação aos elevadores: quantidade de grupos de elevadores no edifício, número de elevadores em cada grupo, velocidade nominal, aceleração, potência, tempo de resposta, distância e velocidade de abertura, tempo de abertura e fechamento das portas, *delay* para o início do fechamento das portas e capacidade de carga que o elevador pode transportar. Tratando-se da capacidade de carga do elevador, outro fator limitante, além do peso, é a sua dimensão. Em hospitais, usualmente é assumido que devem entrar de 1 a 3 visitantes mais 1 a 3 funcionários junto com a maca na cabine, cujas dimensões são de 1.800 mm de largura e 2.800 mm de profundidade. Os elevadores são planejados para serem capazes de transportar todos os

passageiros do hospital em menos de 40 minutos e 25-50% de todas as macas em uma hora. (*Marja-Liisa Siikonen, 2005*)

Ademais, outros critérios também são fundamentais no estudo e importantes quando se diz respeito a eficiência, muitos sistemas têm por objetivo minimizar o tempo de espera do passageiro enquanto outros pretendem reduzir o consumo de energia (*Jianchang Liu, 2007*). Os elevadores são responsáveis por cerca de 4-7% do consumo de energia em um prédio comercial (*Harri Hakala, 2001*), porém considerando o objetivo do presente trabalho, o foco do estudo se direcionará aos critérios de tempo.

Ainda referenciando a pesquisa de *Marja-Liisa Siikonen (2000)*, são levantados como medidas de eficiência os seguintes parâmetros temporais:

- Tempo de espera: tempo contado a partir do momento em que o passageiro aperta o botão e ocorre o *hall call* até o momento em que este entra na cabine;
- Tempo de viagem: começa no momento em que ocorre o *hall call* e termina quando o passageiro sai do elevador após chegar no andar de destino;
- Tempo de chamada: intervalo de tempo entre o instante em que o sistema registra a chamada do passageiro até o instante em que o elevador começa a desacelerar porque está se aproximando ao andar de destino;
- Tempo de resposta do sistema: inicia quando o *hall call* é registrado no sistema e termina quando as portas do elevador começam a se abrir para que o passageiro entre na cabine;
- Tempo de ciclo: tempo médio entre duas partidas consecutivas do elevador (o período em que o elevador está disponível e ninguém entra ou sai é desconsiderado);
- Tempo de ida e volta: consiste no tempo de subida e de descida. O tempo de subida começa quando o elevador começa a subir e termina quando tem seu sentido invertido, o tempo de descida inicia quando o elevador começa a descer e termina quando troca o sentido do seu movimento.

2.1.2. Sistema de Controle de Grupo de Elevadores

Buscando reduzir o tempo de espera pelos elevadores e/ou o tempo de viagem, muitos edifícios utilizam grupos de elevadores, que podem estar atuando individualmente ou podem estar conectados e serem comandados por um sistema de controle único.

Os sistemas mais antigos utilizados para o controle de grupos de elevadores eram baseados no princípio coletivo, no qual os elevadores sempre atendem aos chamados mais próximos de sua localização atual no mesmo sentido do seu deslocamento. A desvantagem

desse algoritmo é o efeito *bunching*, onde vários elevadores se aproximam e competem para atender ao mesmo *hall call.* Isto resulta em muitos elevadores parados no mesmo andar ao mesmo tempo, sendo que, provavelmente, apenas um será, de fato, utilizado e os outros terão realizado uma parada desnecessária, podendo atrasar o atendimento dos demais chamados. (*Aiying Rong, Henri Hakonen e Risto Lahdelma, 2003*)

Em *Muñoz (2006),* um Sistema de Controle de Grupo de Elevadores (EGCSs) é definido como um sistema que gerencia vários elevadores em um edifício para transportar passageiros eficientemente e seu desempenho é medido por meio de diversas métricas, tais como, o tempo médio de espera dos passageiros, a porcentagem de passageiros cujo tempo de espera é superior a um tempo predeterminado, o consumo de energia, entre outros. Basicamente, um EGCS é composto por 4 parâmetros: o conjunto de botões de *hall call*, o conjunto de botões de *cabin call*, o conjunto de elevadores e o controlador do grupo, que irão determinar qual elevador é o mais adequado para atender ao chamado de determinado passageiro. Entretanto esta tarefa é bastante complexa, uma vez que com *n* elevadores e *p* chamadas, existem n^p possibilidades de atendimento e ademais, o sistema precisa lidar com uma série de outros fatores, como o surgimento de novos chamados, a quantidade de pessoas que serão atendidas em um chamado (*Liu*, 2007) e o fluxo de passageiros em um determinado momento.

Diversos algoritmos como rede neural, algoritmos genéricos e/ou *fuzzy logic* podem ser utilizados para resolver esse problema complexo e encontrar uma solução ótima para atribuir um elevador do grupo ao *hall call*. Estes sistemas e métodos podem ser classificados em duas categorias: sistemas baseados no tempo de chegada (*ETA – Estimated Time of Arrival*) e sistemas baseados no destino (*ETD – Estimated Time to Destination* ou *destination dispatch*). (*Rory Smith, Richard Peters, 2008*)

2.1.2.1. Estimated Time of Arrival (ETA)

Em *Elevator Traffic Handbook (Barney, G. C., 2003*), uma das principais características do sistema ETA é a velocidade e continuidade da coleta de dados dos elevadores, registrando os *cabin calls* e *hall calls* e a posição, sentido do deslocamento e *status* de cada elevador, além da estimativa de pessoas embarcadas através de medidas de peso. O registro dessas informações não precisa ser feito a cada décimo de segundo, apenas precisa ser realizado em um intervalo de tempo suficiente para que o sistema realoque os elevadores quando surgir um novo chamado, exceto nos casos em que o elevador já esteja na fase de desaceleração para parar em algum andar.

A alocação dos novos chamados é feita aos elevadores já programados a dirigir-se na mesma direção do chamado ou aos elevadores não programados e que estão desocupados, os demais são considerados “em serviço” e são desconsiderados para a alocação. O *hall call* é atribuído ao elevador que não estiver em serviço e apresentar o menor tempo de viagem.

O tempo de viagem pode ser calculado baseado em uma viagem direta ao pavimento de destino ou considerando paradas intermediárias para atender a *cabin calls* ou *hall calls* no caminho, portanto é importante saber a quantidade de paradas que existem entre o elevador e o andar de destino. O intervalo de viagem é composto por: tempo de viagem entre pavimentos (aceleração, desaceleração, nivelamento e deslocamento na velocidade nominal); tempo de operação da porta (abertura e fechamento); tempo de transferência de passageiros (entrada e saída da cabine); tempo de permanência na porta que possa exceder o tempo de operação das portas.

Não é possível saber quantas pessoas embarcarão no elevador em cada parada, tornando a mensuração do tempo de transferência de passageiros difícil de se avaliar. Se um elevador estiver atendendo a chamados de andares não terminais, como o térreo é, pode-se considerar que se trata de uma situação de tráfego fora do pico. Para esse tipo de situação, é razoável considerar que a relação de *cabin calls* para *hall calls* é de 1,2, ou seja, em 20% dos chamados mais de uma pessoa entra no elevador. *(Barney, G. C., 2003*)

O destino dos passageiros é desconhecido até que estes entrem no elevador, deixando em aberto a possibilidade de haver chamadas com destino entre a posição atual do elevador e o *cabin call* já alocado. Isso faz com que o tempo de viagem de um passageiro sofra alterações após este já ter sido alocado.

Segundo *Aiying Rong, Henri Hakonen e Risto Lahdelma (2003),* o sistema ETA minimiza o tempo de espera dos passageiros no sistema, precisando estimar o número de paradas extras causadas por *hall calls*, além das paradas obrigatórias que já estavam registradas e o pavimento mais provável no qual o elevador irá inverter o sentido do seu movimento. O ETA tenta tratar cada *hall call* igualmente, introduzindo o fator de degradação do sistema para encontrar o elevador apropriado para alocar ao chamado. O sistema não considera apenas o tempo de atendimento do novo *hall call*, mas também o atraso que causará nas demais chamadas que já haviam sido alocadas ao mesmo elevador. Assim sendo, o custo total para a alocação de um novo *hall call* para um elevador i é:

$$t_i^{total} \sum_{j=1}^{n_i} t_{i,j}^{delay} + t_i^{attending}$$

Onde n_i é o número de passageiros que foram alocados ao elevador i mas ainda não foram atendidos, $t_{i,j}^{delay}$ é o tempo que um novo *hall call* irá adicionar ao passageiro j, que foi alocado ao elevador i mas ainda não foi atendido, $t_i^{attending}$ é o tempo estimado para o atendimento do novo *hall call*. Dessa forma, o sistema irá alocar este novo *hall call* ao elevador que possuir o menor custo total.

2.1.2.2. Estimated Time to Destination (Destination Dispatching)

Em *Rory Smith e Richard Peters (2008)*, nos sistemas do tipo *destination dispatching*, os usuários selecionam o seu destino no momento em que realizam o *hall call*. E o sistema, imediatamente, indica qual o elevador que foi alocado para o transporte do passageiro. Além disso, depois de ocorrido a atribuição, o *hall call* não pode ser realocado. E apesar de que os sistemas *destination dispatching* conseguem lidar com até 50% mais tráfego do que os sistemas convencionais, a necessidade de atribuir chamadas imediatamente sem a possibilidade de realocação pode criar ineficiências no sistema.

A empresa ThyssenKrupp, um dos líderes mundiais no segmento de elevadores, criou um sistema de controle de tráfego do tipo ETD (*Estimated Time to Destination*) cujo objetivo é minimizar o tempo total de viagem dos passageiros, ou seja, o intervalo de tempo em que os passageiros estão aguardando a chegada do elevador e estão sendo transportados neste até o pavimento de destino. O sistema faz o cálculo para cada elevador do grupo, assim como calcula o impacto desta nova alocação aos demais chamados já registrados.

Um passageiro realiza um novo *hall call*, que é registrado no sistema com a informação do andar em que o chamado ocorreu e o andar para ao qual o usuário deseja ir. Dessa forma, o sistema calcula o ETD_e, o tempo estimado para o usuário chegar ao destino, em segundos, se o elevador alocado para atender ao passageiro for o e. O intervalo é calculado a partir da determinação do tempo estimado de chegada do elevador e até o andar em que o passageiro o aguarda. Em seguida, é feito o mapeamento da viagem do elevador ao longo do tempo, considerando todas as paradas intermediárias ao longo do percurso do elevador.

Além disso, o sistema também calcula o fator de degradação (*SDF – System Degradation Factor*) do sistema devido a nova alocação para cada elevador do grupo. $SDF_{e,k}$ é o atraso, em segundos, que o novo passageiro irá causar ao passageiro k, se o primeiro for alocado ao elevador e. O SDF é calculado através do mapeamento do percurso do passageiro k antes e depois da introdução do novo passageiro ao sistema. Este fator é considerado para todos os passageiros do sistema, estejam eles esperando pelo elevador, ou já sendo transportados.

Dessa forma, o custo total (*TC – Total Cost*) da alocação do novo passageiro ao elevador e é dado pela soma do fator de degradação de todos os usuários do elevador e e o tempo estimado para a chegada ao destino do novo passageiro. Sendo representado pela seguinte fórmula:

$$TC_e = \sum_{k=1}^{n} SDF_{e,k} + ETD_e$$

Onde TC_e é o custo total do elevador e; k representa cada um dos n passageiros alocados ao elevador e; $SDF_{e,k}$ é o fator de degradação do passageiro k já alocado ao elevador e; e ETD_e é o tempo estimado de chegada ao destino do novo passageiro.

À vista disto, o sistema aloca o novo passageiro ao elevador com o menor custo total. (*Rory Smith* e *Richard Peters – ELEVCON MILAN 2002*)

2.1.2.3. Fuzzy Logic

Um controlador do tipo *Fuzzy Logic* funciona de maneira simples. O sistema recebe uma entrada, admite algum tipo de processamento e devolve uma saída. Basicamente, é composto por quatro componentes: o módulo de difusão, a base de conhecimento, o mecanismo de inferência difusa e o módulo de desfiguração.

O módulo de difusão é um processo em que valores nítidos são a entrada para variáveis linguísticas que serão transformados em valores difusos. Isso envolve o mapeamento dos valores de entrada para refletir seu grau de pertencimento a determinados conjuntos difusos (conjunto com limites claramente definidos que incluem elementos semelhantes, mas que também podem incluir elementos com um grau parcial de pertencimento). O conjunto é definido por funções de associação que determinam como os valores de entrada pertencem a um determinado conjunto, com valores entre 0 e 1. Por exemplo, na lógica Booleana, que admite apenas valores verdadeiro ou falso, na lógica difusa é possível tratar de valores entre verdadeiro ou falso: 0,9 seria algo próximo a verdadeiro. Para o controle do grupo de elevadores, algumas variáveis de entrada poderiam ser distância do elevador, número de paradas ou tempo de resposta do elevador.

A base de conhecimento compreende a base de dados e a base de regras. A função da base de dados é fornecer informações contendo as definições das funções de associações que definem os grupos difusos para o funcionamento adequado do módulo de difusão, do módulo de desfiguração e da base de regras. A base de regras é utilizada para representar o conhecimento especializado em uma estrutura formada a partir de afirmações *IF-THEN*. Cada

afirmação é composta por um antecedente e uma consequência, sendo, normalmente, frases condicionais do tipo *IF* Antecedente *THEN* Consequência. Além disso, também são utilizados conectores *and* (intersecção dos antecedentes) ou *or* (união dos antecedentes) para relacionar dois ou mais antecedentes, formando afirmações como *IF* Antecedente 1 *AND* Antecedente 2 *OR* Antecedente 3 *THEN* Consequência.

A inferência difusa é um método de interpretação dos valores difusos do módulo de difusão e que, baseados na base de regras dos conjuntos difusos, atribui determinados valores aos conjuntos difusos de saída. Durante a inferência, cada regra é comparada com os valores difusos e quando um antecedente é verdadeiro, as consequências são desencadeadas. Esse desencadeamento também é chamado de disparo da regra, e quando disparada, os resultados dos antecedentes são aplicados às consequências de acordo com seu grau de especificação dado pelos seus respectivos antecedentes, formando o conjunto difuso de saída. Essa ação também é conhecida como implicação de regra e é executada para todas as regras disparadas. E todos os seus conjuntos resultantes são agregados e um único conjunto que será processado no módulo de desfiguração, para as saídas finais.

No módulo de desfiguração, o conjunto resultante do processo de inferência é traduzido em valores nítidos com um real significado, o qual ações de controle podem ser tomadas e utilizadas em aplicações reais. (*Jamaludin, J., Rahim, N. A., Hew, W. P., 2009*)

Em *ChangBum Kim et al (1998)*, os autores apresentam um sistema de controle de um grupo de oito elevadores utilizando a *Fuzzy Logic*. O sistema é composto por:

- Geração de estratégias de controle: classifica os dados de tráfego do passageiro em uma das oito categorias que serão apresentados posteriormente, utilizando o método de inferência difusa, e gera uma estratégia de atribuição para o *hall call*.
- Atribuição do *hall call*: atribui o *hall call* ao melhor elevador, considerando as características de todo o grupo, o tráfego de passageiros e a estratégia de controle.
- Gerenciamento de dados: gerencia todas as informações do grupo de elevadores, incluindo os dados dos elevadores, dados do edifício, dados estatísticos, dados de aprendizagem, dados do controle de estratégia e funções de pertencimento da lógica difusa.
- Gerenciamento do elevador: responsável pela comunicação com todos os elevadores e pela coleta de dados de cada um, suas especificações, direção, posição, condições da porta, *status* dos *hall calls* e dos *cabin calls*, e seu peso.

– Gerenciamento do terminal: faz a comunicação do grupo de elevador com o terminal, que gerencia o sistema, recebendo as informações dos *status* dos elevadores, da estratégia de controle, dos dados de tráfego de passageiros e o desempenho do sistema. As configurações do sistema e dos elevadores podem ser alteradas através do terminal.

Na geração de estratégia de controle, o tráfego de passageiros é classificado em oito modos de acordo com suas respectivas características, representados na tabela 1. Essas características podem ser representadas pela tabela 2. Sabe-se que o fluxo de passageiros varia ao longo do dia, como por exemplo, nos horários de entrada e saída dos funcionários de um prédio, ou no horário de almoço. Essa variação exige diferentes estratégias em diferentes horários, como, por exemplo, de manhã muitas pessoas chegam ao prédio para trabalhar e realizam chamadas do térreo para diversos andares e é importante que o tempo de espera dessas pessoas seja minimizado para não haver um grande acúmulo de pessoas. Entretanto, no meio da tarde, o fluxo de passageiros é menor, de forma que se torna interessante minimizar o consumo de energia.

CLASSIFICAÇÃO DO TRÁFEGO DE PASSAGEIROS	
BT	Tráfego total é médio (antes e depois do meio dia)
UP	Muitos passageiros entrando no edifício (início de expediente)
DP	Muitos passageiros saindo do edifício (fim de expediente)
LT-A	Muitos passageiros indo para o restaurante (horário do almoço)
IT	Tráfego total é baixo (de noite)
LT-B	Muitos passageiros voltam do restaurante (fim do horário de almoço)
BTH	Tráfego total é grande (qualquer horário)
HT	Alguns passageiros transitam por entre os diferentes pavimentos (qualquer horário)

Tabela 1 – Classificação do tráfego de passageiros (adaptado de *ChangBum Kim et al, 1998*)

CARACTERÍSTICAS DO TRÁFEGO DE PASSAGEIROS	
UPT	Número de passageiros subindo
DNT	Número de passageiros descendo
CITP	Taxa de passageiros que chegam no andar mais movimentado e vão para os demais andares
DOTP	Taxa de passageiros que chegam em todos os andares com exceção do mais movimentado e vão para todos os andares
TIME	Hora atual

Tabela 2 – Características do tráfego de passageiros (adaptado de *ChangBum Kim et al, 1998*)

Essas características podem ser categorizadas em quantidade de tráfego (TA – *Traffic Amount*) que abrange o UPT e o DNT, taxa de tráfego (TP – *Traffic Percentage*) que abrange a CITP e a DOTP, e tempo (TM – *Time*) que abrange o TIME. Além disso, a figura 2 representa como são definidos os limites dos grupos e suas funções de associação, que determinam a qual classificação pertence determinada variável, ou característica de tráfego.

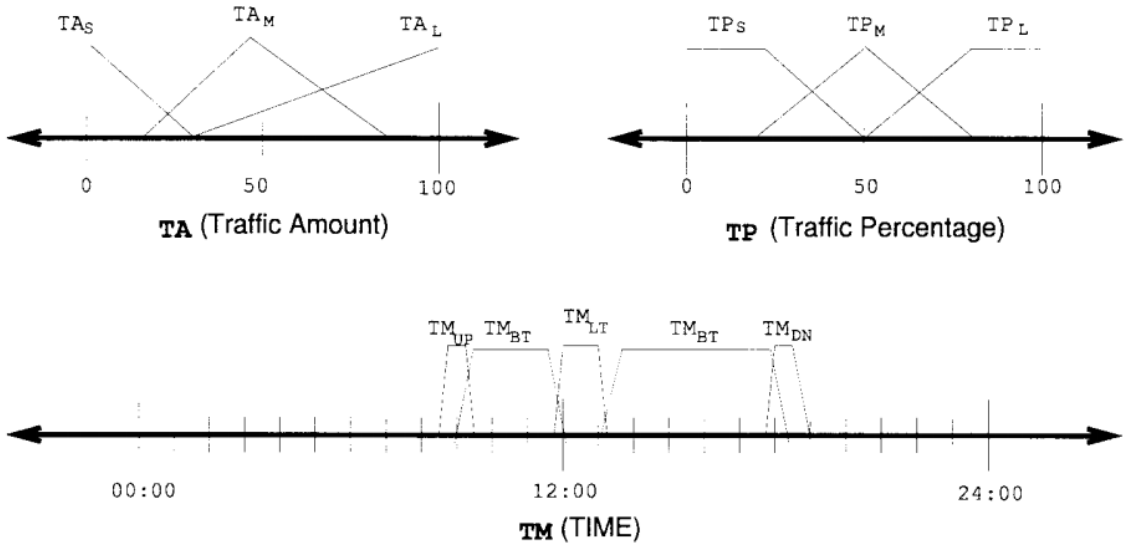


Figura 2 – Funções de associação para as características de tráfego de passageiros (retirado de *ChangBum Kim et al, 1998*)

Onde, TA_S, TA_M e TA_L representam as funções de associação (pequeno, médio e grande – *small, medium, large*) para a quantidade de tráfego (TA), TP_S, TP_M e TP_L representam as funções de associação para a taxa de tráfego (TP) e TM_{UP}, TM_{BT}, TM_{LT} e TM_{DN} representam as funções para o horário (TM).

As cinco características de tráfego são as entradas do método de inferência difusa e os modos de tráfego são as saídas. Alguns exemplos do conjunto da base de regras:

- *If UPT is TA_M and DNT is TA_M and CITP is TP_S and DOTP is TP_M and TIME is TM_{BT} Then belongs to BT.*
- *If UPT is TA_L and DNT is TA_S and CITP is TP_L and DOTP is TP_L and TIME is TM_{UP} Then belongs to UP.*

Para cada modo de tráfego são atribuídos pesos de importância para os seguintes critérios de avaliação: tempo médio de espera (tempo entre o *hall call* e a chegada do elevador, que será chamado de AWT – *Avarage Waiting Time*), porcentagem de passageiros que esperam mais de 60 segundos (chamado de LWP – *Long Waiting Percent*) e o número de movimentos que o elevador realiza em determinada unidade de tempo (o que reflete o consumo estimado de energia, chamado de RNC – *Run Count*). Dessa forma, é definido o objetivo da estratégia de controle, que pode ser, por exemplo, minimizar o tempo de espera ou minimizar o consumo de energia, dependendo do critério considerado mais ou menos importante.

Na etapa de atribuição do *hall call*, são feitas as atribuições dos *hall calls* aos elevadores adequados onde quer que eles ocorram. São realizadas três inferências difusas para testar a adequação de cada um dos elevadores segundo os critérios de avaliação apresentados e também é aplicado um sistema de pesos para determinar qual a adequação geral de cada elevador. O que possuir a maior adequação geral será alocado ao novo *hall call*.

Para determinar a adequação de cada elevador, são consideradas as informações como a direção e o andar dos *hall calls*, condições dos elevadores e futuros *hall calls*. Uma vez que os critérios de avaliação podem ser computados depois que um *hall call* é atribuído a um elevador, os AWT, LWT e RNC podem ser estimados a partir das seguintes variáveis de entrada: tempo de espera do *hall call* (HCWT – *Hall Call Waiting Time*), máximo tempo de espera do *hall call* (maxHCWT – *Maximum Hall Call Waiting Time*), cobertura (CV - *Coverability*, ou capacidade do sistema de controle do grupo de elevadores para alocar futuros *hall calls*) e grau de obtenção (GD – *Gathering Degree*), que é a mínima distância entre um novo *hall call* e um *hall* ou *cabin call* já alocados. Essas variáveis são calculadas para cada elevador quando ocorre um *hall call*, considerando que este tenha sido atribuído a cada um dos elevadores do grupo.

Tendo essas variáveis definidas, são realizadas as inferências difusas para cada um dos critérios de avaliação, de forma a determinar a adequação de cada elevador. Alguns exemplos da base de regras dessas inferências são:

- *If HCWT is $HCWT_S$ and CV is CV_L Then S_{AWT} is LARGE.*
- *If maxHCWT is $maxHCWT_S$ and CV is CV_L Then S_{LWP} is LARGE.*

– *If maxHCWT is* $maxHCWT_L$ *and GD is* GD_S *Then* S_{RNC} *is SMALL.*

Onde, S_{AWT}, S_{LWP} e S_{RNC} representam a adequação do elevador segundo os critérios de avaliação de tempo de espera, porcentagem de passageiros que esperam mais de 60 segundos e número de movimentos que o elevador realiza, respectivamente, e *SMALL, MEDIUM* e *LARGE* são os termos do domínio que define a adequação dos elevadores, representados na figura 3. A partir das adequações encontradas, são aplicados os pesos para cada critério, resultando na adequação geral de cada elevador, que será alocado para atender ao novo *hall call*.

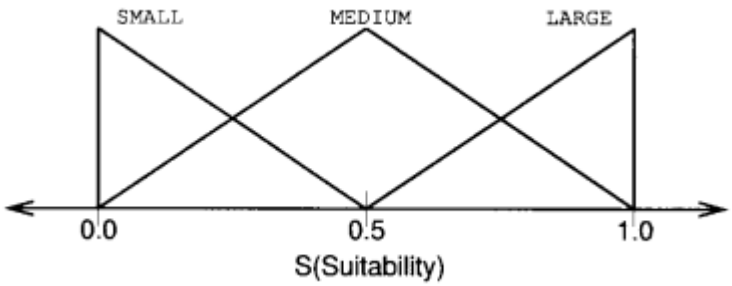


Figura 3 – Representação dos termos do domínio da adequação (retirado de *ChangBum Kim et al, 1998*)

2.1.2.4. Comparação entre os sistemas

Para melhor compreensão da diferença entre os sistemas apresentados, em *Rory Smith* e *Richard Peters (ELEVCON MILAN 2002)* é apresentado uma comparação entre a alocação que seria feita para o seguinte caso: um grupo de três elevadores, alguns chamados já registrados no sistema e um novo *hall call* é realizado no sétimo andar.

Utilizando o Sistema ETA

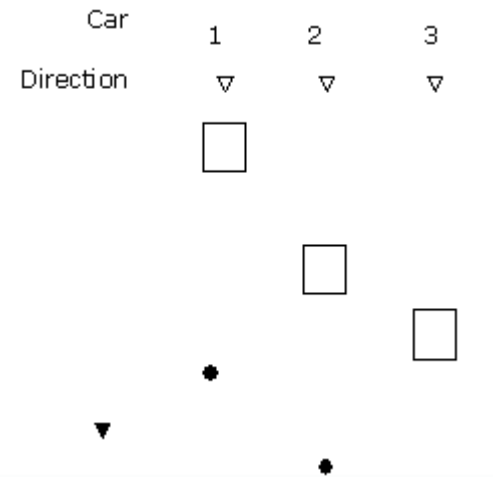


Figura 4 – Utilizando o Sistema ETA

- Elevador 1: está a 15s do novo chamado. Ele deve parar no oitavo andar, o que irá atrasar o elevador em 10s em sua jornada ao sétimo andar.
- Elevador 2: está a 10s do chamado.
- Elevador 3: está a 5s do chamado.

Dessa forma, o sistema ETA irá alocar o elevador 3 ao novo *hall call*, que possui o menor tempo estimado de espera.

Utilizando Fuzzy Logic

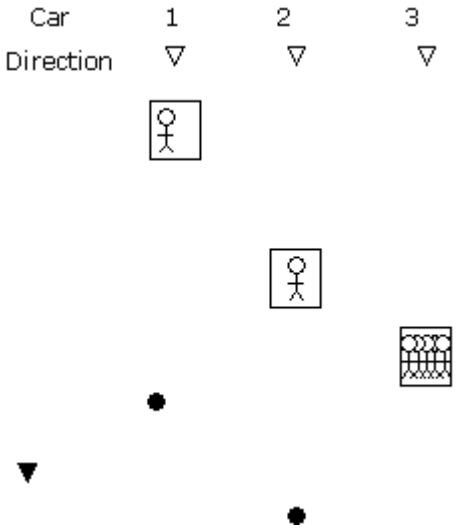


Figura 5 – Utilizando Fuzzy Logic

Existem diversas variáveis que podem ser consideradas no cálculo, porém uma possível decisão poderia ser a seguinte:

- Elevador 1: está longe e está quase vazio.
- Elevador 2: está perto e está quase vazio.
- Elevador 3: é o elevador mais próximo e está quase cheio.

Dessa forma, o elevador 2 possui preferência em relação ao elevador 3 que está praticamente cheio, uma vez que este tipo de sistema não considera apenas o menor tempo de espera, mas também leva em conta outras variáveis como o tempo dos demais passageiros ou a capacidade. Atrasar um elevador que está praticamente cheio representa um atraso maior quando somado o atraso de cada passageiro. Além disso, há a possibilidade de o elevador estar totalmente cheio e realizar uma parada desnecessária em um andar onde o passageiro não conseguiria entrar.

Utilizando o Sistema ETD

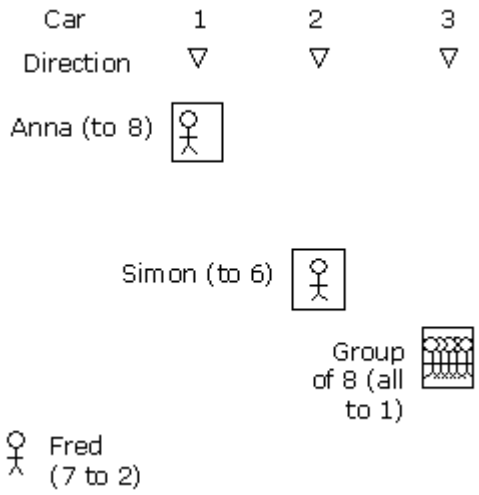


Figura 6 – Utilizando o Sistema ETD

Considerando que Fred está alocado no elevador 1 e deseja ir do sétimo ao segundo andar:

- O atraso no tempo da Anna é de 0s.
- Fred deve esperar 15s para o elevador chegar ao seu andar, mais 10s para o desembarque de Anna.
- Uma vez embarcado, a viagem de Fred até o segundo andar é de 25s.
- O custo total será de 50s.

Considerando que Fred está alocado no elevador 2 e deseja ir do sétimo ao segundo andar:

- A viagem de Simon será atrasada em 10s, para o embarque do Fred no sétimo andar.
- Fred deve esperar 10s para poder embarcar.
- Fred leva 25s para chegar ao seu destino, mais 10s para desembarcar Simon.
- O custo total será de 55s.

Considerando que Fred está alocado no elevador 3 e deseja ir do sétimo ao segundo andar:

- Um grupo de 8 pessoas sofrerá um atraso de 10s para embarcar o Fred e 10s para desembarca-lo.
- Fred deve esperar 5s para poder embarcar.
- Uma vez embarcado, Fred levará 25s para chegar ao seu destino no segundo anda.
- O custo total será de 190s.

Segundo o sistema ETD, o elevador 1 será alocado para atender Fred.

2.1.2.5. Estratégias de Controle

Em *Crites* e *Barto (1998)*, são apresentas algumas estratégias de controle que podem ser utilizadas em conjunto com os sistemas expostos no presente trabalho.

Abordagem de zoneamento: a cada elevador é atribuído uma zona do edifício. Ele irá atender apenas aos *hall calls* dentro da sua zona e ficará estacionado dentro dela enquanto estiver ocioso. O objetivo da abordagem de zoneamento é manter os carros razoavelmente bem separados e, assim, manter os tempos de espera baixo. Se possível, é atribuído a um *hall call* um elevador que já iria parar naquele andar devido a um *cabin call*. Caso contrário, um elevador alocado à zona em que está o *hall call* irá atendê-lo, de forma que o tempo de espera é menor se comparado ao que o passageiro aguardaria se fosse esperar por um elevador localizado em andares fora da zona. A divisão das zonas é um parâmetro de controle que afeta o tempo médio de espera e o consumo de energia.

Abordagens baseadas em regras: sistemas convencionais costumam utilizar uma fórmula fixa de avaliação baseada nas posições atuais dos elevadores e nas localizações atuais dos chamados. Entretanto, existem sistemas que consideram as futuras posições dos elevadores e as probabilidades de um novo *hall call* surgir em determinado andar, antes mesmo de terem sido realizados. Por exemplo, uma regra de controle poderia ser *IF* existe um *hall call* registrado em um andar acima *AND* há um grande número de elevadores indo se movimentando em direção aos andares superiores *THEN* atribuir um dos elevadores que estão subindo baseado no tempo estimado de espera.

Outras abordagens heurísticas: o algoritmo LQF (*Longest Queue First*) atribui os elevadores com movimento ascendente ao andar com o maior tempo de espera e o algoritmo HUFF (*Highest Unanswered Floor First*) aloca os elevadores com movimento ascendente aos andares em que há um maior número de passageiros esperando. Os dois algoritmos são projetados para tráfegos menores, que não estão em horário de pico, e atribuem os elevadores em movimento descendente a todos os *hall calls* que ainda não foram atendidos. O algoritmo DLB (*Dynamic Load Balancing*) tenta manter os elevadores espaçados uniformemente, atribuindo setores contíguos e não sobrepostos a cada elevador, de maneira a equilibrar suas cargas, reatribuindo os setores após cada evento.

2.2. Modelagem Matemática

A modelagem matemática é utilizada por muitos estudiosos para representar sistemas reais a fim de prever o seu comportamento, de maneira simplificada, porém o mais próximo da realidade possível.

Klüber e Burak (2005) estudaram a concepção de modelagem matemática de alguns autores, buscando interpretações para a educação matemática voltada ao desenvolvimento da modelagem. Em seu estudo foram levantados os conceitos do próprio Burak e de Biembengut, apresentados a seguir.

Segundo Burak (1992), a modelagem matemática é um "conjunto de procedimentos cujo objetivo é construir um paralelo para tentar explicar, matematicamente, os fenômenos presentes no cotidiano do ser humano, ajudando-o a fazer predições e a tomar decisões". O autor descreve a modelagem em cinco etapas: 1. Escolha do tema; 2. Pesquisa exploratória; 3. Levantamento dos problemas; 4. Resolução dos problemas e o desenvolvimento do conteúdo matemático no contexto do tema; 5. Análise crítica das soluções.

Em seu livro Modelagem Matemática & Implicações no Ensino-Aprendizagem de Matemática, Biembengut (1999) define a modelagem matemática como "o processo que envolve a obtenção de um modelo", interligando matemática e realidade. A professora apresenta alguns procedimentos os quais a modelagem segue: 1. Interação (reconhecimento da situação problema e familiarização com o assunto a ser modelado); 2. Matematização (formulação e resolução do problema em termos matemáticos – onde é feita a "tradução" do problema para a linguagem matemática, identificando as constantes envolvidas e selecionando as variáveis para descrever suas relações em termos matemáticos); 3. Modelo matemático (interpretação da solução e validação do modelo, verificando o nível de aproximação que este tem do problema apresentado).

Kai Velten (2009) diz que a dificuldade dos problemas tratados pela ciência e pela engenharia é, tipicamente, dada pela complexidade dos sistemas em questão e a modelagem providencia uma ferramenta adequada para romper essa complexidade e tornar o problema acessível. O autor define modelo como "*para um observador B, um objeto A* é um modelo de um objeto A na medida em que B pode usar A* para responder questões que lhe interessam sobre A*".

E o estudo desses sistemas complexos pode ser dividido nas seguintes etapas:

- Definições: definição do problema a ser resolvido ou da questão a ser respondida; definição do sistema, a parte da realidade pertinente ao problema.
- Análise dos sistemas: identificação das partes do sistema que são relevantes para o problema.
- Modelagem: desenvolvimento de um modelo do sistema baseado nos resultados da etapa anterior.

– Simulação: aplicação do modelo ao problema; derivação de uma estratégia para resolver o problema.

– Validação: verificação se a estratégia derivada da etapa anterior realmente resolve o problema.

Segundo *Harry Perros (2009)*, um modelo pode ser classificado em icônico, análogo ou simbólico. Um modelo icônico é composto pela réplica exata das propriedades do sistema real em menor escala, como por exemplo modelos de aviões, maquetes ou mapas. Um modelo análogo utiliza um conjunto de propriedades para representar um sistema real, esse tipo de modelo pode utilizar um sistema para representar outro sistema, como por exemplo, um sistema hidráulico pode ser utilizado para representar analogamente um sistema elétrico ou um sistema de tráfego. E um modelo simbólico representa as propriedades de um sistema real por meio de símbolos, como equações matemáticas ou programas de computação. Modelos de simulação são classificados como modelos simbólicos.

Os modelos de simulação ainda podem ser classificados como modelos determinísticos (que não possuem elementos probabilísticos, como programação linear ou não linear e programação dinâmica) ou modelos estocásticos (que contém elementos probabilísticos, como teoria das filas, processos estocásticos e modelos de confiabilidade).

Além disso, os modelos de simulação são compostos por um sistema, um conjunto de entidades que são logicamente relacionadas e de interesse para uma aplicação específica. As entidades são:

– Ambiente: cada sistema pode ser visto como um subsistema de um sistema mais abrangente.

– Interdependência: nenhuma atividade ocorre em total isolamento.

– Subsistemas: cada sistema pode ser dividido em subsistemas.

– Organização: praticamente todos os sistemas consistem em elementos ou componentes organizados, os quais interagem entre si para desempenhar a função do sistema.

– Mudança: a atual condição ou estado do sistema varia ao longo do tempo. (*Harry Perros, 2009*)

No geral, um modelo de simulação serve para quantificar o desempenho de um sistema de acordo com os diferentes valores atribuídos para os parâmetros de entrada. Estes parâmetros, também chamados de variáveis, podem ser divididos em dois grupos: variáveis incontroláveis, que são dadas pelas propriedades do sistema e não podem ser manipuladas ou modificadas, e

variáveis controláveis, que podem ser modificadas de modo a encontrar a solução para o problema proposto.

Além disso, as variáveis também podem ser classificadas em exógenas ou endógenas. As variáveis exógenas são aquelas que não dependem de outras variáveis, ou seja, seu valor não se altera quando outras variáveis admitem diferentes valores. Alguns exemplos são o intervalo de tempo entre duas chegadas sucessivas ao sistema, tempo de atendimento de um cliente, número de atendentes ou regras de priorização. Por outro lado, as variáveis endógenas são aquelas cujo valor é determinado por outras variáveis ao longo da simulação, como o tempo de espera médio na fila (que pode sofrer influência do tempo de atendimento, por exemplo) ou a média do tamanho de uma fila (influenciada pela taxa de chegada e pelo tempo de atendimento).

2.2.1. Principais Paradigmas da Simulação

Em seu estudo, *Borschev e Filippov (2004)* apresentam os três maiores paradigmas da simulação: Sistemas Dinâmicos, Eventos Discretos e Modelos Baseados em Agentes. Porém, antes de entrar nos detalhes de cada técnica de simulação, é importante entender os níveis de abstração em um modelo de simulação. *Borschev e Filippov* exemplificam sistemas de controle e simulações de micro tráfego (que consideram o fluxo de cada objeto individualmente) como modelos com baixo nível de abstração pois é importante que os objetos de estudo tenham tamanhos exatos e que as distâncias, velocidades e tempos sejam conhecidos. Com um nível um pouco maior de abstração, os autores citam modelos de chão de fábrica e modelos de logística de armazéns com transportes e operações de carga e descarga, pois possuem certo grau de abstração em suas trajetórias físicas e utilizam o tempo médio em seus cálculos.

Modelos de tráfego e de transporte em nível macro não podem considerar veículos e objetos individualmente, eles trabalham com volumes de agentes, caracterizando um nível de abstração médio. E por fim, as simulações com maior grau de abstração são aquelas abordadas em termos de valores agregados, *feedbacks* globais e tendências, com raciocínios do tipo “se o número de empregos aumentar, então haverá aumento da imigração”.

E os autores também mostram que para cada nível de abstração, um determinado paradigma de simulação pode ser mais adequado ou não. A figura 7, a seguir, mostra alguns exemplos de modelos de simulação e seu nível de abstração e a figura 8 mostra quais paradigmas são mais adequados de acordo com o nível de abstração do problema.

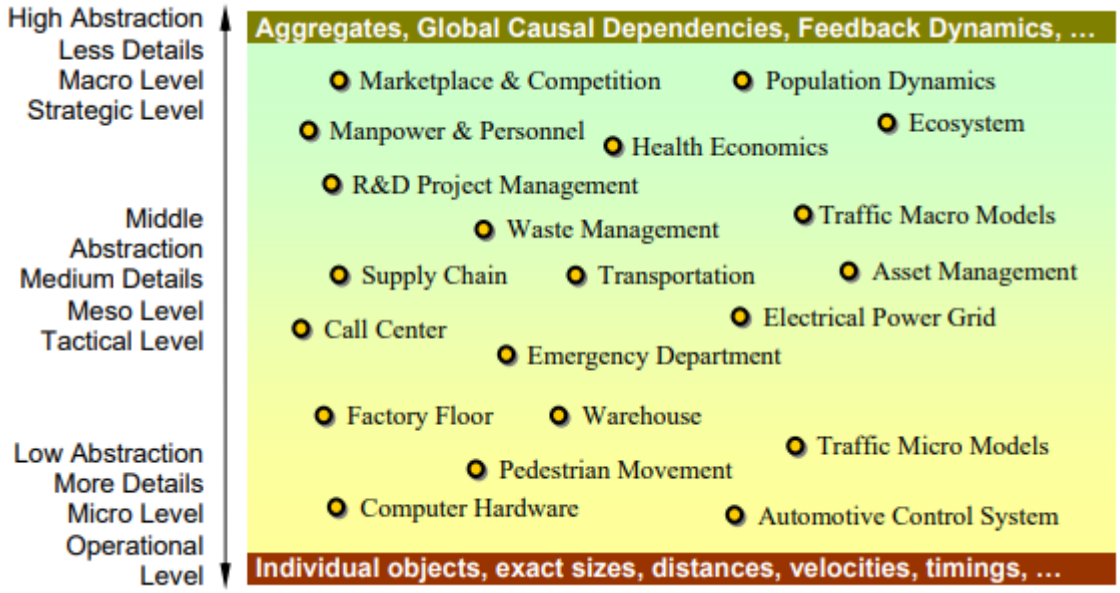


Figura 7 – Nível de abstração de exemplos de modelos de simulação

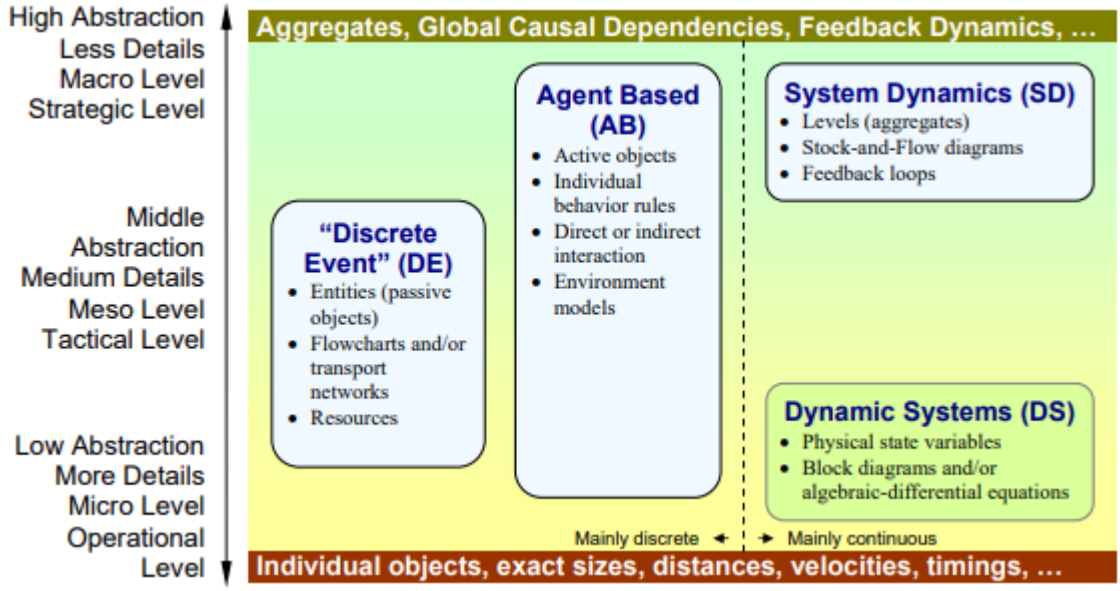


Figura 8 – Paradigmas mais adequados para diferentes níveis de abstração

2.2.1.1. Sistemas Dinâmicos

Segundo *Forrester (1950)*, os sistemas dinâmicos são o estudo das características das informações e *feedbacks* da atividade industrial para mostrar como a estrutura organizacional, suas políticas de decisão e tempos interagem e influenciam no sucesso da empresa. Os processos do mundo real são representados em termos de "estoques" (como por exemplo, de material, conhecimento, pessoas ou dinheiro), fluxos entre esses estoques e informações que

determinam os valores dos fluxos. E enquanto o modelo estiver funcionando com base em agregados, os itens do estoque são indistinguíveis, ou seja, não possuem individualidade

Os sistemas dinâmicos abstraem eventos e entidades individuais concentrando-os em visões agregadas com foco em políticas de decisão. Para tanto, é necessário descrever o comportamento do sistema como uma série de *loops* e *feedbacks* interagindo, equilibrando ou se reforçando. Além disso, o modelador precisa pensar em termos de dependências estruturais globais e fornecer dados quantitativos e precisos.

2.2.1.2. Eventos Discretos

A modelagem por eventos discretos é baseada na representação de sistemas reais através do conceito de entidades, recursos e blocos, descrevendo o fluxo das entidades do sistema e os recursos compartilhados. Entidades são objetos que representam pessoas, documentos, tarefas, etc., que viajam por entre os blocos do fluxograma que representam o conjunto de processos pelos quais estas devem passar. Durante o percurso, as entidades ficam em filas, são processadas, ocupam e liberam recursos, etc.

De maneira geral, a modelagem por eventos discretos pode ser considerada como a definição de um algoritmo global de processamento das entidades, normalmente com processos estocásticos, ou seja, através de variáveis aleatórias que representam a evolução do sistema ao longo do tempo. (*Forrester, 1950*)

2.2.1.3. Modelos Baseados em Agentes

Ao contrário dos outros dois paradigmas, modelos baseados em agentes são descentralizados, ou seja, não é possível definir um comportamento global para todos as entidades (nesse caso, chamadas de agentes). Cada agente tem seu comportamento definido individualmente, seguindo suas próprias regras, vivendo em conjunto e interagindo com outros agentes e com o ambiente.

Esse tipo de modelo também permite um estudo da população de agentes, cujo comportamento global pode surgir a partir da análise do comportamento de diversos agentes. (*Forrester, 1950*)

3. METODOLOGIA

3.1. Metodologia de Pesquisa

3.1.1. Apresentação da Metodologia Escolhida

Segundo *Robson Bonomo* (professor da Universidade Federal do Espírito Santo), uma pesquisa científica possui alguns critérios que a classificam de acordo com seus objetivos, apresentados na tabela 3.

CRITÉRIOS	**CLASSIFICAÇÕES**
Finalidade	Básica Aplicada
Objetivos	Exploratória Descritiva Explicativa
Procedimentos	Bibliográfica Documental Experimental Outros
Natureza	Qualitativa Quantitativa
Local de realização	Campo Laboratorial

Tabela 3 – Classificação dos tipos de pesquisa (adaptado do *professor Robson Bonomo*)

A finalidade de uma pesquisa científica pode ser básica, tendo por objetivo a satisfação do desejo de adquirir conhecimentos, sem que haja uma aplicação prática prevista, ou aplicada, no qual os conhecimentos adquiridos são utilizados para aplicação prática voltados para a solução de problemas concretos da vida moderna. A presente pesquisa se enquadra como aplicada, pois tem por finalidade propor uma solução para melhoria da eficiência do sistema de elevadores do ICESP.

De acordo com os seus objetivos, uma pesquisa pode ser exploratória, sendo caracterizada como um primeiro contato com o tema para conhecer os fatos e fenômenos relacionados através de levantamentos bibliográficos, entrevistas com profissionais da área e visitas a instituições e empresas. Também pode ser descritiva, levantando características conhecidas sobre o fato e/ou processo na forma de levantamentos ou observações sistemáticas

do processo escolhido. Ou explicativa, visando explicar e criar uma teoria a respeito de um processo ou fenômeno, aprofundando o conhecimento da realidade e tendo uma grande preocupação com a identificação dos fatores que determinam a ocorrência ou a forma como o processo ocorre. Essa pesquisa é classificada como descritiva pois visa a simulação do sistema de elevadores do ICESP através da coleta de dados já existentes a respeito da demanda pelos elevadores, assim como da maneira como estas são atendidas.

Os procedimentos de coleta de dados podem ser do tipo pesquisa experimental (consiste em fazer experiências, no qual o processo estudado é reproduzido de forma controlada com o objetivo de descobrir os fatore que o produzem ou que por ele são produzidos), *ex-post-facto* (o pesquisador não tem controle direto sobre as variáveis independentes pois suas manifestações já ocorreram e são intrinsecamente não manipuláveis, dessa forma, são feitas inferências sobre as relações entre as variáveis em observações direta, a partir da variação concomitante entre as variáveis dependentes e independentes), levantamento (caracteriza-se pela interrogação direta de pessoas cuja opinião deseja-se conhecer e é dividida nas seguintes etapas: seleção da amostra, aplicação de questionários, formulários ou entrevistas, tabulação dos dados e análise com auxílio de ferramentas estatísticas), estudo de caso (estudo aprofundado e exaustivo de um ou de poucos objetos, de maneira a permitir o seu conhecimento amplo e detalhado), pesquisa-ação (pesquisa social cujo objetivo é a resolução de um problema coletivo e no qual os pesquisadores e os participantes representativos estão envolvidos de modo cooperativo) ou pesquisa documental (coleta de dados de fontes de informações ainda não publicadas e que não foram analisadas). O presente estudo utiliza o procedimento *ex-post-facto* pois será baseado na coleta das informações das viagens que já foram realizadas pelo sistema de elevadores.

Quanto a sua natureza, uma pesquisa pode ser quantitativa, que, segundo *Wainer* (2007), vem da tradição das ciências naturais, onde as variáveis são objetivas (ou seja, diferentes pesquisadores obterão os mesmos resultados em suas observações) e medidas em escalas numéricas, ou qualitativa, vinda das ciências sociais e cujas variáveis não podem ser medidas, apenas observadas de maneira subjetiva. A natureza dessa pesquisa é quantitativa, uma vez que são tratados dados numéricos como, por exemplo, o tempo de viagem de um passageiro do elevador.

E por fim, o local de realização da pesquisa pode ser no campo ou laboratório. No campo, o estudo é feito onde acontece o fato ou fenômeno através da coleta de dados e observação de fenômenos "*in natura*", podendo ser feita por observação direta, levantamento ou estudo de caso. No laboratório, é caracterizado pela inferência artificial na produção do fato/processo ou pela artificialização do ambiente e dos mecanismos de percepção para que o

fato ou processo seja percebido adequadamente, permitindo o estabelecimento de padrões desejáveis de observação, a captação de dados para descrição e análise e o controle do fato/processo/fenômeno. O presente estudo é realizado no campo, pois é realizado com o estudo de caso do sistema de elevadores do ICESP.

3.2. Metodologia do Projeto

3.2.1. Fonte de Dados

Os dados utilizados no presente estudo foram fornecidos pela equipe de segurança do ICESP através de planilhas em Excel com as informações sobre os andares em que ocorreram os *hall calls* e *cabin calls* dos seis elevadores centrais do prédio no dia 05 de outubro de 2020, no período das 6:00 às 22:00.

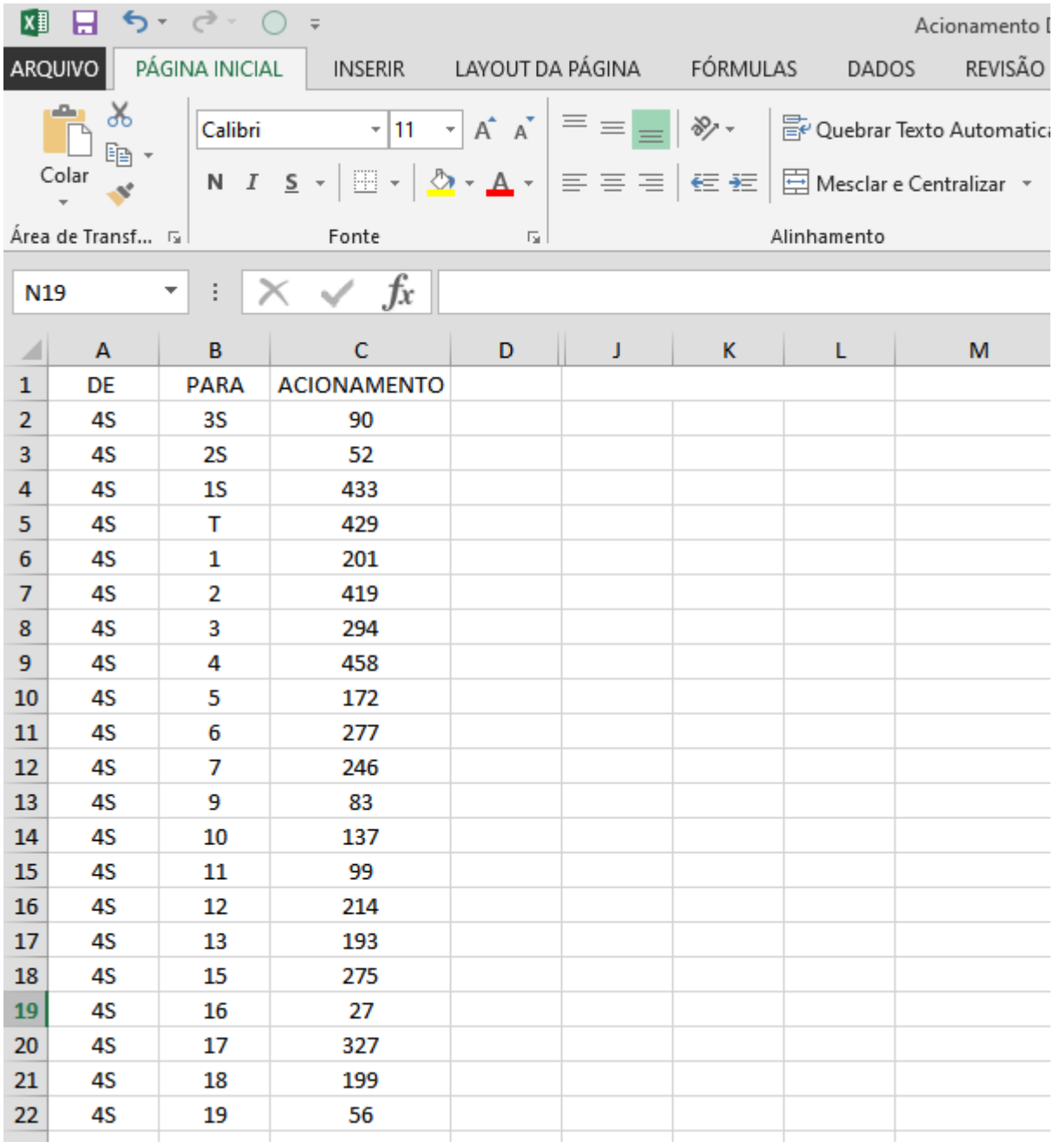
Também foram apresentados os dados sobre o número médio de pessoas que circulam o instituto por dia, tempo de abertura e fechamento das portas, velocidade média dos elevadores, altura dos pavimentos, tempo de espera médio pelos elevadores e tempo médio de viagem. Estes dados respondidos pela mesma equipe, por e-mail.

3.2.1.1. Descrição das Variáveis de Estudo

As variáveis de estudo podem ser divididas em *inputs* e *outputs*, ou seja, os dados de entrada e parâmetros do modelo e dados de saída e resultados da simulação.

Os *inputs* foram determinados a partir das necessidades da simulação para que o modelo pudesse se aproximar do sistema de elevadores real, de maneira que depois foram solicitadas e fornecidas pela equipe de segurança do ICESP.

A distribuição de *hall calls* e *cabin calls* pelos andares do prédio são de extrema importância para a determinação da política de tomada de decisões do sistema de elevadores mais adequada para a demanda de passageiros do ICESP. Entretanto, foi disponibilizado apenas a quantidade de acionamentos (quantidade de vezes que um elevador foi solicitado e/ou direcionado) dos seis elevadores centrais. Esses dados foram fornecidos no formato de uma tabela com a contagem de acionamentos realizados de um determinado andar para outro ao longo de um dia, em um formato “DE PARA”.



	A	B	C	D	J	K	L	M
1	DE	PARA	ACIONAMENTO					
2	4S	3S	90					
3	4S	2S	52					
4	4S	1S	433					
5	4S	T	429					
6	4S	1	201					
7	4S	2	419					
8	4S	3	294					
9	4S	4	458					
10	4S	5	172					
11	4S	6	277					
12	4S	7	246					
13	4S	9	83					
14	4S	10	137					
15	4S	11	99					
16	4S	12	214					
17	4S	13	193					
18	4S	15	275					
19	4S	16	27					
20	4S	17	327					
21	4S	18	199					
22	4S	19	56					

Figura 9 – Tabela de acionamentos dos elevadores do ICESP no dia 05 de outubro (fonte: fornecido pela equipe de segurança do ICESP)

Além disso, alguns valores constantes também foram disponibilizados para que o modelo se aproximasse da realidade, estes são apresentados na tabela 4.

DADOS	**VALORES**
Quantidade de andares atendidos pelos elevadores centrais	27 andares
Quantidade de elevadores do sistema	6 elevadores
Número médio de pessoas que entram no ICESP por dia	8.500 pessoas por dia
Tempo de abertura das portas dos elevadores	4 segundos
Tempo de fechamento das portas dos elevadores	4 segundos
Velocidade média dos elevadores	4 metros por segundo
Altura dos pavimentos	4,35 metros

Tabela 4 – Dados constantes do ICESP (fonte: conversa com equipe de segurança do ICESP)

Os *outputs* resultantes foram os tempos de espera pelo elevador e os tempos de permanência do sistema, contados a partir da chegada do passageiro no sistema e sua saída após chegar no andar de destino, em segundos. Também foram analisadas as taxas de ocupação dos elevadores e o tamanho das filas de espera de cada andar. Esses indicadores foram utilizados como parâmetros de comparação para as diferentes políticas de tomada de decisão do sistema de elevadores, objetivo do presente estudo.

3.2.1.2. Restrições dos Dados

A base de dados fornecida leva em consideração o número de acionamentos realizados, ou seja, quantas vezes um elevador foi solicitado no terceiro andar com destino ao oitavo, por exemplo. O que abre margem para divergências em relação ao número de pessoas que realmente utilizaram o elevador, uma vez que um passageiro pode estar impaciente e solicitar o elevador mais de uma vez ou um passageiro pode pegar "carona" e entrar no elevador sem ter realizado um *hall call* ou um *cabin call*.

Além disso, as informações foram apresentadas como um resumo dos acontecimentos ocorridos no dia 05 de outubro, não considerando as variações de demanda ao longo do dia (horários de pico como entrada, horário de almoço e saída de funcionários). Ademais, também não foram considerados as variações ao longo da semana, o que seria interessante analisar, considerando que existem alguns serviços do instituto que são prestados em dias específicos da semana e influenciam na demanda pelos elevadores.

3.2.2. Ferramentas de Estudo Utilizadas

Para a realização do estudo foram utilizadas diferentes ferramentas. A análise dos dados foi feita utilizando o Microsoft Excel, uma vez que as informações foram disponibilizadas por

meio de planilhas e os resultados são gerados em um arquivo .txt que pode facilmente ser aberto em planilhas do Excel. Além disso, a complexidade da análise dos dados não exigiu nenhuma ferramenta em específico.

A simulação foi feita com o *software* AnyLogic, que permite a representação de sistemas reais através de simulações baseadas em agentes, eventos discretos e dinâmica de sistemas. Em conjunto, foi utilizado o Microsoft Excel para tratamento dos dados de saída do sistema e validação da simulação, através da geração de histogramas e cálculos de média, por exemplo, e códigos em linguagem de programação Python para tratamento dos dados da taxa de ocupação dos elevadores e tamanho das filas de cada andar do prédio.

3.2.2.1. AnyLogic

O AnyLogic é uma ferramenta de modelagem de simulação desenvolvida pela empresa The AnyLogic Company. A empresa teve início em 1990, em uma época onde havia grande interesse na abordagem matemática para modelagem e simulação de processos paralelos. Um grupo de pesquisa chamado The Distributed Computer Network desenvolveu um *software* para a análise da exatidão de programas, permitindo a notação gráfica para estrutura e comportamento do sistema. O sucesso da pesquisa inspirou o grupo a desenvolver um *software* de simulação focado em métodos aplicados: simulação, análise de desempenho, comportamento de sistemas estocásticos, otimização e visualização, e que contava com diversos recursos da tecnologia da informação: abordagem orientada a objetos, uso do Java, interface gráfica moderna, entre outros.

Os modelos de simulação do *software* podem se basear em qualquer um dos três paradigmas de simulação descritos na revisão bibliográfica (eventos discretos, dinâmica de sistemas e modelagem baseada em agentes) ou na combinação entre eles. No presente estudo, foram utilizadas as técnicas de eventos discretos, uma vez que se trata da representação do fluxo de passageiros e elevadores e o estudo do comportamento do sistema segundo diferentes políticas de tomada de decisão do sistema, onde seu nível de abstração é baixo e as informações sobre o sistema são mais exatos, como o número de andares do prédio e a velocidade média dos elevadores.

Além disso, o AnyLogic permite ao usuário refinar seus modelos de simulação utilizando código Java, ampliando o potencial da modelagem e auxiliando na análise dos dados de saída. Esse recurso foi bastante utilizado no estudo, permitindo a personalização das diferentes políticas de decisão do sistema de elevadores e também a extração dos resultados para sua validação com o auxílio do Microsoft Excel.

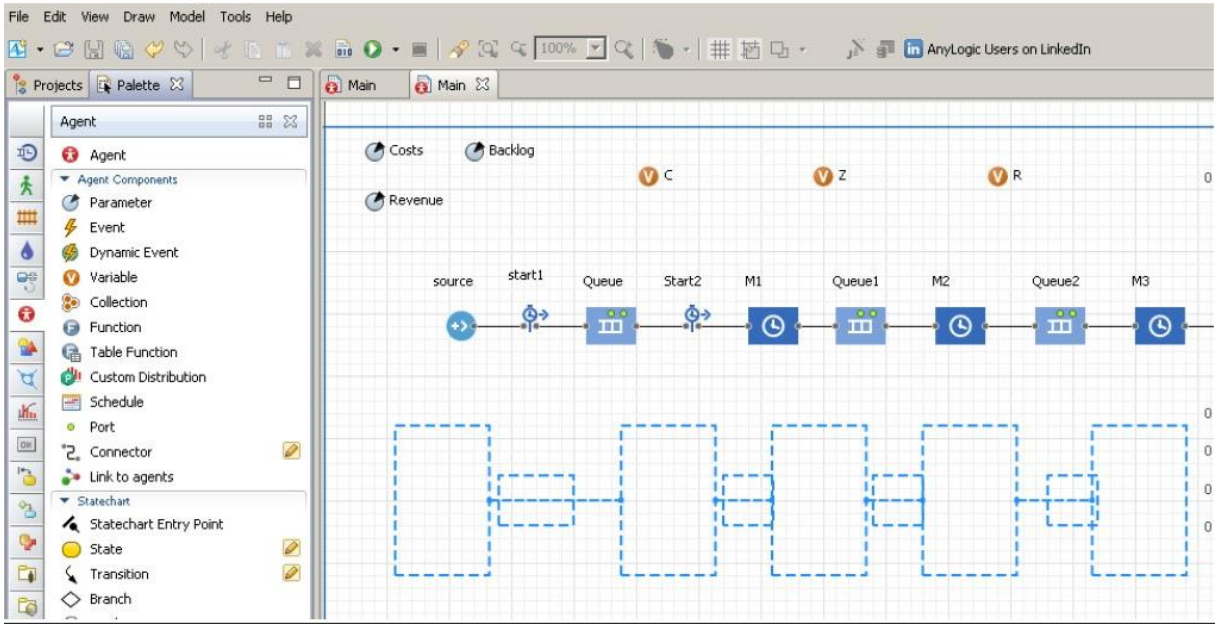


Figura 10 – *Software* AnyLogic (retirado no site do AnyLogic)

4. PROJETO

4.1. Elaboração do modelo de simulação

Inicialmente, foi construído um modelo genérico que não levava em consideração as características do ICESP, mas representava todos os processos pelos quais um elevador passa. Vale ressaltar que, na entrada e saída de passageiros, ocorrem as etapas de abertura das portas, entrada e/ou saída dos passageiros e fechamento das portas antes que o elevador volte a se movimentar.

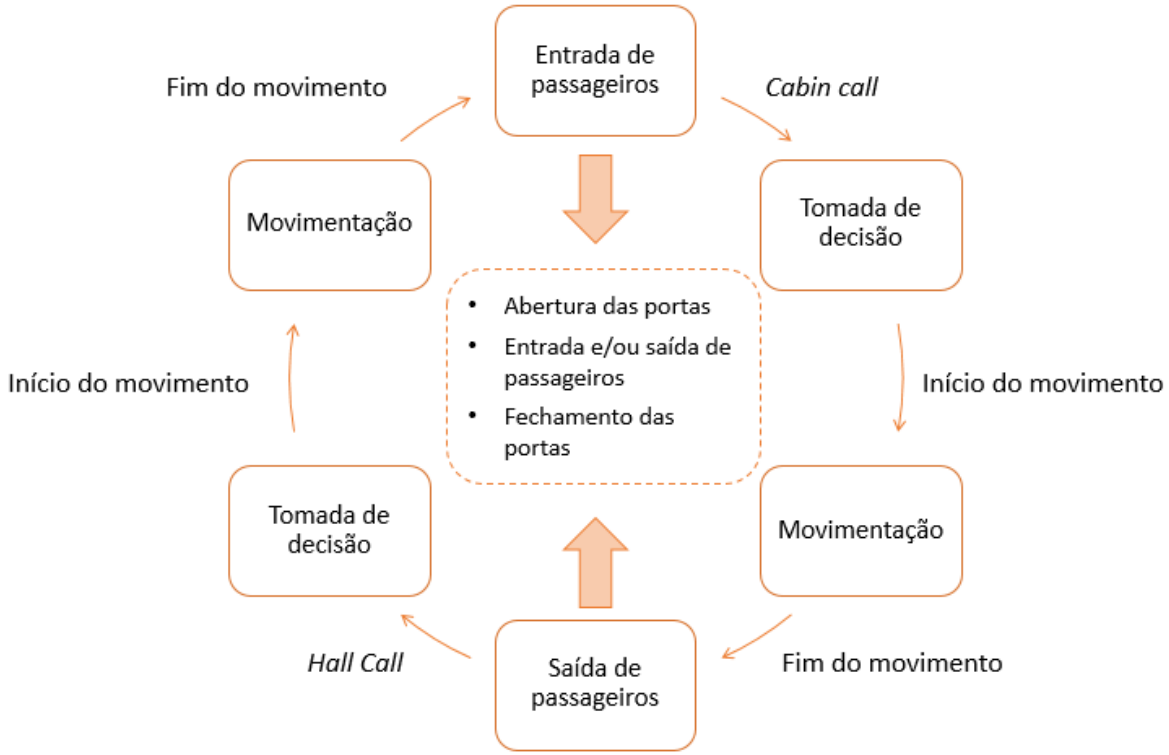


Figura 11 – Fluxo de processos do elevador

Outro ponto que deve ser ressaltado é que durante a etapa de saída de passageiros em um determinado andar, outras pessoas podem entrar no mesmo andar, de maneira que não irão ocorrer as etapas de tomada de decisão e movimentação antes da entrada de novos passageiros. Tendo isso em mente, a modelagem do fluxo de acontecimentos pode ser melhor representada na figura 12.

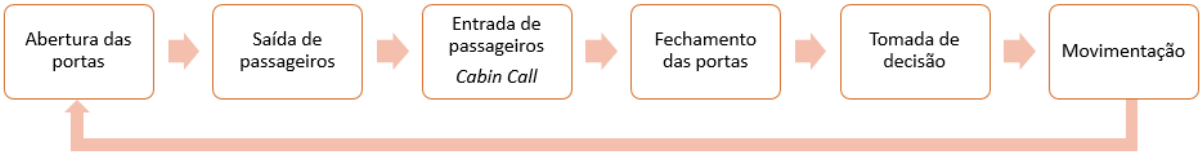


Figura 12 – Fluxo de processos dos elevadores simplificado

A etapa de tomada de decisão corresponde ao momento em que o sistema de elevadores deve decidir em qual sequência os *hall calls* e *cabin calls* deverão ser atendidos, dado que durante o fluxo vários chamados podem ocorrer e estes nem sempre serão atendidos na mesma ordem em que ocorreram.

4.1.1. Estrutura do modelo

O modelo de simulação é constituído por blocos interligados que representam o fluxo de processos pelos quais os elevadores passam, como por exemplo, um bloco de *delay* que pode representar o tempo de abertura das portas, ou seja, quando o componente do elevador passar por esse bloco, ele será mantido por um tempo pré-determinado, que representa o tempo para as portas do elevador serem abertas, antes de seguir o fluxo para o próximo bloco.

Além disso, em cada bloco também é possível determinar funções em linguagem Java que complementam os seus respectivos papéis na simulação. Um exemplo é a utilização do código, na imagem a seguir, para atualizar o número de passageiros dentro do elevador quando os passageiros saem deste.

On dropoff:

```
if (elevadores.get(container.elevadorIndex).numeroPessoas > 0) {
    elevadores.get(container.elevadorIndex).numeroPessoas = elevadores.get(container.elevadorIndex).numeroPessoas - 1;
    saidas = saidas+1;
}
```

Figura 13 – Exemplo de função em Java para complemento das funções de um determinado bloco

Neste tópico serão apresentados os blocos utilizados na modelagem do sistema de elevadores e no seguinte, as premissas consideradas para cada bloco, como o tempo de abertura das portas, por exemplo, e suas respectivas funções complementares.

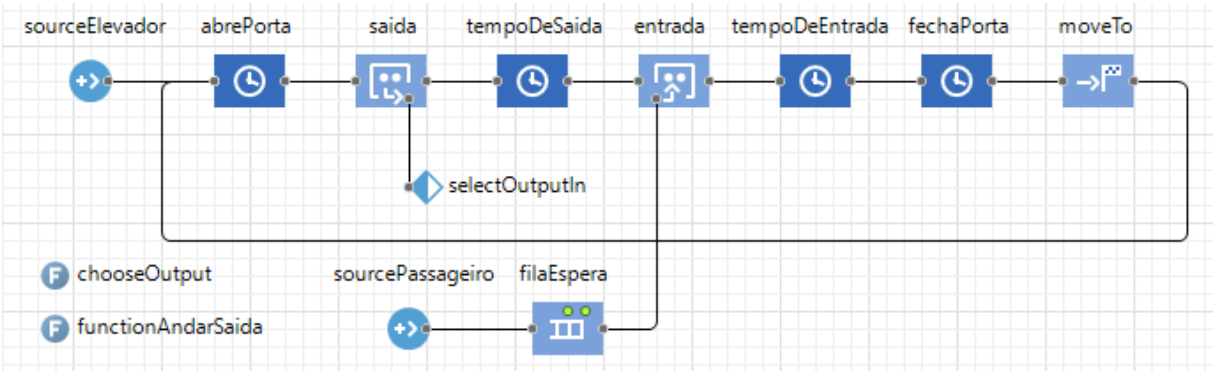


Figura 14 – Blocos interligados representando o fluxo de processos pelos quais os elevadores passam

No bloco *sourceElevador* são criados os objetos que representam os elevadores e seus respectivos argumentos (ou variáveis) que os caracterizam. Os elevadores são apenas inicializados neste bloco e depois não voltam a passar por lá durante seu fluxo.

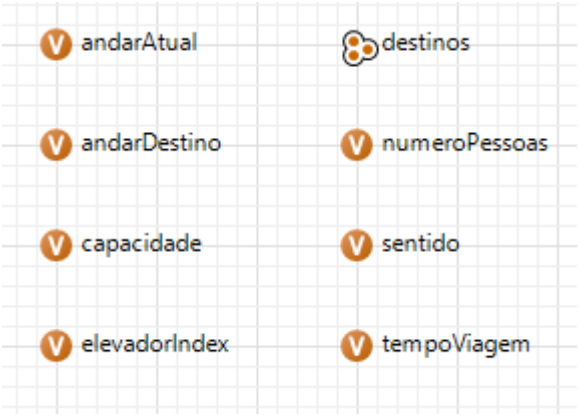


Figura 15 – Argumentos que caracterizam os elevadores

Na figura 15 são apresentadas as variáveis que caracterizam os elevadores, sendo elas:

- *andarAtual*: armazena a informação do andar em que o elevador se encontra parado;
- *andarDestino*: armazena a informação sobre para qual andar o elevador deve se dirigir;
- *capacidade*: representa o número máximo de passageiros que podem estar dentro do elevador ao mesmo tempo;
- *elevadorIndex*: índice que indica qual o elevador que está passando por determinado bloco;
- *destinos*: trata-se de uma lista com os andares de destino os quais o elevador irá atender em uma determinada ordem;
- *numeroPessoas*: representa o número de pessoas que há dentro do elevador;
- *sentido*: representa o sentido para o qual o elevador está indo, podendo assumir os valores 1 (elevador subindo) ou –1 (elevador descendo);
- *tempoViagem*: representa o tempo de viagem do elevador, ou seja, o tempo que ele irá demorar para ir do andar atual para o andar de destino. Esta variável é calculada a partir da diferença entre o andar de destino e o andar atual multiplicada pelo tempo que o elevador demora para se locomover entre dois andares consecutivos.

No bloco *sourcePassageiro* são criados os passageiros que irão embarcar nos elevadores, ou seja, representa a chegada de pessoas que irão utilizar o elevador no sistema. Assim como os elevadores, os passageiros também possuem variáveis que determinam suas características:

Figura 16 – Argumentos que caracterizam os passageiros

- *andarEntrada*: variável que armazena o andar em que o passageiro chegou;
- *andarSaida*: variável que armazena o andar em que o passageiro irá desembarcar do elevador;
- *passageiroIndex*: índice que indica qual o passageiro que está passando por determinado bloco;
- *tempoEntrada*: variável que armazena o momento (em segundos) da simulação em que o passageiro entrou na fila de espera pelo elevador;
- *tempoSaida*: variável que armazena o momento (em segundos) da simulação em que o passageiro saiu da fila de espera pelo elevador;
- *diferencaTempo*: diferença entre o tempo de saída e o tempo de entrada na fila de espera, representando o período de tempo o qual o passageiro passou esperando na fila até poder embarcar em um dos elevadores.

O bloco *filaEspera* representa a fila de espera dos passageiros que estão aguardando a chegada de um elevador para embarcarem neste.

Os blocos *abrePorta* e *fechaPorta* representam os tempos de abertura e fechamento das portas dos elevadores, respectivamente. Nestes blocos, o elevador espera um determinado tempo antes de seguir para o próximo bloco.

Assim como os blocos de abertura e fechamento das portas, os blocos *tempoDeSaida* e *tempoDeEntrada* também fazem com que os elevadores que passarem por estes blocos esperem um determinado tempo antes de seguirem o fluxo dos processos. Nestes casos, eles representam os tempos que o elevador deve permanecer com as portas abertas para que os passageiros entrem e/ou saiam do elevador.

O bloco *entrada* representa a entrada de passageiros no elevador, o qual possui a condição de que um passageiro só pode entrar no elevador se este não estiver com sua ocupação máxima e só é permitida a entrada de passageiros que possuam o andar de entrada igual ao andar atual do elevador.

De maneira análoga, o bloco *saida* representa a saída de passageiros do elevador e possui a condição de que os passageiros saem se o andar atual do elevador for o mesmo que o seu andar de saída. Além disso, o bloco *saida* é conectado ao bloco *selectOutputIn*, que chama a função *chooseOutput*, a seguir, determinando o andar de saída na representação dos andares do prédio.

```
if (agent.andarSaida == -4) {
	return saida4S;
} else if (agent.andarSaida == -3) {
	return saida3S;
} else if (agent.andarSaida == -2) {
	return saida2S;
} else if (agent.andarSaida == -1) {
	return saida1S;
} else if (agent.andarSaida == 0) {
	return saida0;
} else if (agent.andarSaida == 1) {
	return saida1;
} else if (agent.andarSaida == 2) {
	return saida2;
} else if (agent.andarSaida == 3) {
	return saida3;
} else if (agent.andarSaida == 4) {
	return saida4;
} else if (agent.andarSaida == 5) {
	return saida5;
} else if (agent.andarSaida == 6) {
	return saida6;
} else if (agent.andarSaida == 7) {
	return saida7;
} else if (agent.andarSaida == 8) {
	return saida8;
} else if (agent.andarSaida == 9) {
	return saida9;
} else if (agent.andarSaida == 10) {
	return saida10;
} else if (agent.andarSaida == 11) {
	return saida11;
} else if (agent.andarSaida == 12) {
	return saida12;
} else if (agent.andarSaida == 13) {
	return saida13;
} else if (agent.andarSaida == 14) {
	return saida14;
} else if (agent.andarSaida == 15) {
	return saida15;
} else if (agent.andarSaida == 16) {
	return saida16;
} else if (agent.andarSaida == 17) {
	return saida17;
} else if (agent.andarSaida == 18) {
	return saida18;
} else if (agent.andarSaida == 19) {
	return saida19;
} else if (agent.andarSaida == 20) {
```

```
        return saida20;
} else if (agent.andarSaida == 21) {
        return saida21;
} else {
        return saida22;
}
```

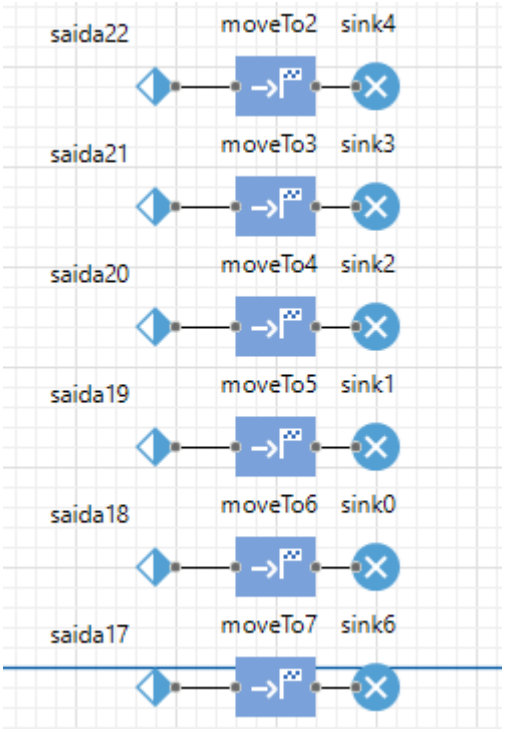


Figura 17 – Representação dos andares do prédio (andares: 17, 18, 19, 20, 21 e 22)

Os andares são representados pelos blocos da imagem acima, onde cada conjunto de três blocos representa um andar. Dentre esses três blocos, o *saidaXX* representa a saída da função *chooseOutput* que direciona o passageiro para o andar de saída correto de acordo com sua respectiva variável pré-definida. O bloco *moveTo* é o bloco responsável pelo movimento da representação gráfica dos passageiros. Este bloco será melhor detalhado a seguir, quando for tratado o bloco *moveTo* do fluxo do elevador. E o bloco *sinkX* é onde os passageiros deixam o sistema.

Por fim, o bloco *moveTo* é responsável pela movimentação dos elevadores na representação gráfica da simulação. No presente modelo, os elevadores são representados por retângulos que percorrem as representações dos andares do prédio e o bloco é responsável pela movimentação desses retângulos, fazendo-os subir ou descer de acordo com seus andares atuais e de destino.

4.1.2. Premissas consideradas e funções complementares em cada bloco

No bloco *sourceElevador* é determinado o número máximo de elevadores que irão fazer parte da simulação e o local onde serão inicializados, que são seis elevadores e andar térreo, respectivamente. Além disso, também é determinado a sua velocidade de movimentação, 4 metros por segundo, conforme informado pela equipe de segurança do ICESP.

E por fim, são atribuídos os valores dos índices de cada elevador, segundo o código, em linguagem de programação Java, a seguir:

```
agent.elevadorIndex = (int)sourceElevador.count();
```

Estes assumem valores de 0 a 5, de acordo com a ordem com a qual foram criados, sendo que 0 representa o primeiro elevador e 5 o sexto e último.

No bloco *sourcePassageiro* foram determinados o número máximo de passageiros que deve entrar no sistema e sua taxa de chegada. De acordo com os dados fornecidos, chegam em média 8500 passageiros por dia. E os dados fornecidos foram retirados entre 6:00 e 22:00, o que resulta em uma taxa de chegada de passageiros de aproximadamente 8,854 passageiros por minuto.

Além disso, neste bloco são determinados os andares de chegada e saída dos passageiros, calculados através de probabilidades (apresentadas no tópico 4.2 Tratamento Inicial dos Dados de Entrada), advindos do tratamento dos dados recebidos do ICESP e aplicados no modelo de simulação através do código a seguir.

```
Random random = new Random();
double numero = random.nextDouble();

if(numero <= 0.037504) {
	agent.andarEntrada = -4;}
if(numero > 0.037504 && numero <= 0.087425) {
	agent.andarEntrada = -3;}
if(numero > 0.087425 && numero <= 0.121739) {
	agent.andarEntrada = -2;}
if(numero > 0.121739 && numero <= 0.176965) {
	agent.andarEntrada = -1;}
if(numero > 0.176965 && numero <= 0.222872) {
	agent.andarEntrada = 0;}
if(numero > 0.222872 && numero <= 0.263077) {
	agent.andarEntrada = 1;}
if(numero > 0.263077 && numero <= 0.304929) {
	agent.andarEntrada = 2;}
if(numero > 0.304929 && numero <= 0.364490) {
	agent.andarEntrada = 3;}
if(numero > 0.364490 && numero <= 0.404779) {
```

```
       agent.andarEntrada = 4;}
if(numero > 0.404779 && numero <= 0.431952) {
       agent.andarEntrada = 5;}
if(numero > 0.431952 && numero <= 0.469567) {
       agent.andarEntrada = 6;}
if(numero > 0.469567 && numero <= 0.500133) {
       agent.andarEntrada = 7;}
if(numero > 0.500133 && numero <= 0.524591) {
       agent.andarEntrada = 8;}
if(numero > 0.524591 && numero <= 0.552923) {
       agent.andarEntrada = 9;}
if(numero > 0.552923 && numero <= 0.584354) {
       agent.andarEntrada = 10;}
if(numero > 0.584354 && numero <= 0.619847) {
       agent.andarEntrada = 11;}
if(numero > 0.619847 && numero <= 0.647788) {
       agent.andarEntrada = 12;}
if(numero > 0.647788 && numero <= 0.689787) {
       agent.andarEntrada = 13;}
if(numero > 0.689787 && numero <= 0.694045) {
       agent.andarEntrada = 14;}
if(numero > 0.694045 && numero <= 0.731116) {
       agent.andarEntrada = 15;}
if(numero > 0.731116 && numero <= 0.768652) {
       agent.andarEntrada = 16;}
if(numero > 0.768652 && numero <= 0.805376) {
       agent.andarEntrada = 17;}
if(numero > 0.805376 && numero <= 0.842782) {
       agent.andarEntrada = 18;}
if(numero > 0.842782 && numero <= 0.874785) {
       agent.andarEntrada = 19;}
if(numero > 0.874785 && numero <= 0.910712) {
       agent.andarEntrada = 20;}
if(numero > 0.910712 && numero <= 0.950491) {
       agent.andarEntrada = 21;}
if(numero > 0.950491 && numero <= 1) {
       agent.andarEntrada = 22;}

agent.andarSaida = functionAndarSaida(agent.andarEntrada);
```

Inicialmente, o código gera um número aleatório entre 0 e 1 que, dependendo do intervalo em que este se encontra, é determinado o andar de chegada de um passageiro, conforme a tabela 5. Esta tabela mostra a probabilidade acumulada do andar de chegada de um determinado passageiro.

Andar de entrada	**Probabilidade**	**Probabilidade acumulada**	**Intervalo para determinação do andar de entrada**
4S (-4)	0,037504	0,037504	$X \leq 0,037504$
3S (-3)	0,049921	0,087425	$0,037504 < X \leq 0,087425$
2S (-2)	0,034314	0,121739	$0,087425 < X \leq 0,121739$
1S (-1)	0,055226	0,176965	$0,121739 < X \leq 0,176965$
Térreo (0)	0,045908	0,222872	$0,176965 < X \leq 0,222872$
1	0,040205	0,263077	$0,222872 < X \leq 0,263077$
2	0,041852	0,304929	$0,263077 < X \leq 0,304929$
3	0,059561	0,364490	$0,304929 < X \leq 0,364490$
4	0,040289	0,404779	$0,364490 < X \leq 0,404779$
5	0,027173	0,431952	$0,404779 < X \leq 0,431952$
6	0,037615	0,469567	$0,431952 < X \leq 0,469567$
7	0,030566	0,500133	$0,469567 < X \leq 0,500133$
8	0,024458	0,524591	$0,500133 < X \leq 0,524591$
9	0,028332	0,552923	$0,524591 < X \leq 0,552923$
10	0,031431	0,584354	$0,552923 < X \leq 0,584354$
11	0,035493	0,619847	$0,584354 < X \leq 0,619847$
12	0,027941	0,647788	$0,619847 < X \leq 0,647788$
13	0,041999	0,689787	$0,647788 < X \leq 0,689787$
14	0,004258	0,694045	$0,689787 < X \leq 0,694045$
15	0,037071	0,731116	$0,694045 < X \leq 0,731116$
16	0,037636	0,768752	$0,731116 < X \leq 0,768752$
17	0,036624	0,805376	$0,768752 < X \leq 0,805376$
18	0,037406	0,842782	$0,805376 < X \leq 0,842782$
19	0,032003	0,874785	$0,842782 < X \leq 0,874785$
20	0,035926	0,910712	$0,874785 < X \leq 0,910712$
21	0,039779	0,950491	$0,910712 < X \leq 0,950491$
22	0,049509	1,000000	$0,950491 < X \leq 1,000000$

Tabela 5 – Probabilidade acumulada dos andares de chegada dos passageiros

Sendo definido o andar de chegada, o andar de saída é definido pela função chamada na última linha do código acima (*functionAndarSaida*). Esta função trabalha de maneira

semelhante à determinação do andar de entrada, porém, para cada andar de entrada, são definidas diferentes probabilidades acumuladas para cada andar de saída. A função de determinação do andar de saída dos passageiros recebe o andar de entrada e retorna o andar de saída, sendo apresentada no anexo 1.

E por fim, neste bloco também são atribuídos os índices para cada passageiro, conforme o código:

```
agent.passageiroIndex = (int)sourcePassageiro.count();
```

No bloco *filaEspera*, é feito o controle da lista dos *hall calls*. Para cada passageiro que entrar na fila de espera, é armazenado seu andar de entrada na lista "chamados" para que possam ser atendidos pelo sistema de elevadores. Na entrada da fila também é atribuído à variável *tempoEntrada* o momento em que o passageiro entrou na fila. Essas duas atribuições são feitas através do código a seguir.

```
chamados.add(agent.andarEntrada);
agent.tempoEntrada = time(SECOND);
```

Na saída do bloco (saída da fila), é retirado o *hall call* do passageiro que saiu da fila da lista "chamados", uma vez que seu chamado já foi atendido. E também é atribuído à variável *tempoSaida* o momento em que o passageiro saiu da fila. Dessa forma, é possível calcular o tempo de permanência do passageiro na fila, por meio da diferença entre os tempos de saída e entrada na fila de espera. Os códigos utilizados neste bloco são apresentados a seguir.

```
for(int i = 0; i < chamados.size(); i++) {
	if (agent.andarEntrada == chamados.get(i)) {
		chamados.remove(i);
		i = i - 1;
	}
}
```

```
agent.tempoSaida = time(SECOND);
agent.diferencaTempo = agent.tempoSaida - agent.tempoEntrada;
```

Para os blocos *abrePorta* e *fechaPorta* foi determinado o tempo de quatro segundos para abertura e fechamento das portas, respectivamente, também informado pela equipe de segurança do instituto.

Nos blocos *tempoDeSaida* e *tempoDeEntrada* foi determinado um tempo de 2 segundos para cada passageiro que for entrar ou sair do elevador. Ou seja, o elevador irá demorar dois segundos vezes o número de passageiros que entraram e/ou saíram do elevador para seguir para o próximo bloco do fluxo.

Como citado no tópico anterior, o bloco *entrada* possui duas condições para que um passageiro possa entrar no elevador, a ocupação, ou a variável *numeroPessoas*, do elevador não ter atingido sua capacidade de 10 passageiros e o andar de entrada do passageiro ser igual ao andar atual do elevador. Para tal, foi utilizada uma variável auxiliar "cont" que assume dois valores: 1, caso a ocupação do elevador for menor do que 10 e 0, caso sua ocupação chegue em 10. Dessa maneira, a condição de entrada é determinada pela sentença:

```
cont == 1 && elevadores.get(container.elevadorIndex).andarAtual ==
agent.andarEntrada
```

No bloco *entrada* também é feito o controle da lista de destinos de cada elevador, no qual para cada passageiro que entra no elevador é adicionado seu andar de destino na lista *destinos* e também é somado 1 ao número de passageiros presentes dentro do elevador. Este controle é realizado pelo seguinte código:

```
if (elevadores.get(container.elevadorIndex).numeroPessoas < container.capacidade)
{
	cont = 1;
	elevadores.get(container.elevadorIndex).destinos.add(agent.andarSaida);
	elevadores.get(container.elevadorIndex).numeroPessoas =
elevadores.get(container.elevadorIndex).numeroPessoas + 1;
	entradas = entradas + 1;
}
else {cont = 0;}


temp.clear();
if (elevadores.get(container.elevadorIndex).destinos.size() > 0) {
	a = -4;
	b = 24;
	while (a <= 23) {
		for (int i = 0; i <
elevadores.get(container.elevadorIndex).destinos.size(); i++) {
			if (elevadores.get(container.elevadorIndex).destinos.get(i) ==
a && elevadores.get(container.elevadorIndex).destinos.get(i) != b) {
				temp.add(a);
				b = a;
			}
		}
		a = a+1;
	}
```

```
}

elevadores.get(container.elevadorIndex).destinos.clear();
for(int k = 0; k < temp.size(); k++) {
	elevadores.get(container.elevadorIndex).destinos.add(temp.get(k));
}
```

No bloco *saida*, a condição de saída dos passageiros é realizada pelo código:

```
agent.andarSaida == elevadores.get(container.elevadorIndex).andarAtual
```

Além disso, neste bloco também é feita a atualização do número de passageiros dentro do elevador de acordo com a quantidade de passageiros que o deixaram e a atualização da lista dos andares de destino do elevador. As atualizações são feitas pelo código:

```
if (elevadores.get(container.elevadorIndex).numeroPessoas > 0) {
	elevadores.get(container.elevadorIndex).numeroPessoas =
elevadores.get(container.elevadorIndex).numeroPessoas - 1;
	saidas = saidas+1;
}
```

```
if (elevadores.get(container.elevadorIndex).destinos.size() > 0) {
	for (int i = 0; i <
elevadores.get(container.elevadorIndex).destinos.size(); i++) {
		if (elevadores.get(container.elevadorIndex).destinos.get(i) ==
elevadores.get(container.elevadorIndex).andarAtual) {
			elevadores.get(container.elevadorIndex).destinos.remove(i);
		}
	}
}
```

No bloco *moveTo*, é determinado o andar para o qual o elevador será direcionado, podendo ser um *hall call* da lista *chamados* ou um *cabin call* da lista *destinos*. No caso de ser atendido um *cabin call*, a sequência a qual os andares da lista de destino serão atendidos será definida pela política de decisão estipulada no estudo.

Além disso, neste bloco também é determinado o tempo de viagem, de acordo com a variável *tempoViagem*, cujo cálculo já foi apresentado no tópico 4.1.1. Estrutura do modelo e o código que o determina é o seguinte:

```
elevadores.get(agent.elevadorIndex).tempoViagem =
abs(elevadores.get(agent.elevadorIndex).andarDestino -
elevadores.get(agent.elevadorIndex).andarAtual)*1.0875;
```

Onde 1,0875 é o tempo, em segundos, que um elevador leva para percorrer um andar, calculado a partir da divisão da altura de cada pavimento, 4,35 metros, pela velocidade média dos elevadores, 4 metros por segundo. Estes dados foram fornecidos pela equipe de segurança dos ICESP.

4.1.3. Políticas de decisão

No presente estudo foram comparadas três políticas de tomada de decisão a respeito da ordem de atendimento dos *cabin calls* que compõe a lista *destinos* de cada elevador.

A primeira política determina que os andares de destino serão atendidos seguindo a ordem do andar mais baixo, ou seja, o próximo andar a ser atendido pelo elevador será o andar mais baixo da lista *destinos*. O código que o determina é o seguinte:

```
elevadores.get(agent.elevadorIndex).andarDestino =
elevadores.get(agent.elevadorIndex).destinos.get(0);
```

A segunda política determina que o elevador só mudará o sentido de seu movimento quando não houver mais andares para serem atendidos, se mantido o seu sentido atual. Ou seja, se o elevador estiver subindo, este atenderá primeiro todos os *cabin calls* acima de seu andar atual, em ordem crescente. Se estiver descendo, atenderá a todos os *cabin calls* abaixo de seu andar atual, em ordem decrescente. Esta política é definida pelo seguinte código:

```
if (elevadores.get(container.elevadorIndex).destinos.size() > 0) {
	if(elevadores.get(container.elevadorIndex).sentido == 1) {
		for (int i = 0; i <
elevadores.get(container.elevadorIndex).destinos.size(); i++) {
			if (elevadores.get(container.elevadorIndex).destinos.get(i) >
elevadores.get(container.elevadorIndex).andarAtual) {
				aux = 1;
			}
			else {
				aux = -1;
			}
		}
	}

	if(elevadores.get(container.elevadorIndex).sentido == -1) {
		for (int i = elevadores.get(container.elevadorIndex).destinos.size()
- 1; i >= 0; i--) {
			if (elevadores.get(container.elevadorIndex).destinos.get(i) <
elevadores.get(container.elevadorIndex).andarAtual) {
				aux = -1;
			}
```

```
                    else {
                           aux = 1;
                    }
             }
       }
       elevadores.get(container.elevadorIndex).sentido = aux;
}


if(elevadores.get(agent.elevadorIndex).sentido == 1) {
       for (int i = 0; i < elevadores.get(agent.elevadorIndex).destinos.size();
i++) {
             if (elevadores.get(agent.elevadorIndex).destinos.get(i) >
elevadores.get(agent.elevadorIndex).andarAtual) {
                    elevadores.get(agent.elevadorIndex).andarDestino =
elevadores.get(agent.elevadorIndex).destinos.get(i);
                    break;
             }
       }
}
else {
       for (int i = elevadores.get(agent.elevadorIndex).destinos.size() - 1; i >=
0; i--) {
             if (elevadores.get(agent.elevadorIndex).destinos.get(i) <
elevadores.get(agent.elevadorIndex).andarAtual) {
                    elevadores.get(agent.elevadorIndex).andarDestino =
elevadores.get(agent.elevadorIndex).destinos.get(i);
                    break;
             }
       }
}
```

E a terceira política faz com que os elevadores atendam ao andar da lista de destinos mais próximo ao seu andar atual. Sendo determinados pelo código a seguir.

```
aux = 100;
for (int i = 0; i < elevadores.get(agent.elevadorIndex).destinos.size(); i++) {
       if (abs(elevadores.get(agent.elevadorIndex).andarAtual -
elevadores.get(agent.elevadorIndex).destinos.get(i)) < aux) {
             aux = abs(elevadores.get(agent.elevadorIndex).andarAtual -
elevadores.get(agent.elevadorIndex).destinos.get(i));
             elevadores.get(agent.elevadorIndex).andarDestino =
elevadores.get(agent.elevadorIndex).destinos.get(i);
       }
}
```

4.1.5. Validação do modelo

Para validar o modelo, foram elaboradas duas tabelas semelhantes as que foram feitas com os dados de entrada e saída de passageiros no tópico 4.2 – Tratamento Inicial dos Dados de Entrada, porém com os dados resultantes da simulação. A partir disso, foi feita a comparação entre as probabilidades de cada andar ser o andar de entrada e/ou saída de um passageiro.

Essa comparação foi feita com o cálculo do MAPE (*Mean Absolute Percentage Error*), realizado nas três rodadas de simulação (uma para cada política de decisão, mas para as quais as probabilidades de cada andar ser a entrada ou a saída de um passageiro são as mesmas para todas as políticas) resultando nos erros absolutos médios percentuais apresentados na tabela a seguir. Vale ressaltar que para todas as rodadas de simulação foi considerada uma amostra de aproximadamente 50 mil passageiros.

Política de decisão	**MAPE médio dos andares de entrada**	**MAPE médio dos andares de saída**
Política 1	1,90%	7,97%
Política 2	2,39%	8,39%
Política 3	2,03%	8,35%

Tabela 6 – MAPE médio para cada uma das três políticas de decisão

4.2. Tratamento Inicial dos Dados de Entrada

A partir da tabela de dados “DE PARA” com as informações sobre os acionamentos dos elevadores disponibilizada para o estudo, foram construídas duas tabelas.

A primeira contém as probabilidades de cada pavimento ser o andar de chegada de um passageiro, ou seja, mostra a probabilidade de um passageiro entrar no sistema no primeiro andar, por exemplo.

ANDAR DE ENTRADA	PROBABILIDADE
4S (-4)	0,037504
3S (-3)	0,049921
2S (-2)	0,034314
1S (-1)	0,055226
Térreo (0)	0,045908
1	0,040205
2	0,041852
3	0,059561
4	0,040289
5	0,027173
6	0,037615
7	0,030566
8	0,024458
9	0,028332
10	0,031431
11	0,035493
12	0,027941
13	0,041999
14	0,004258
15	0,037071
16	0,037636
17	0,036624
18	0,037406
19	0,032003
20	0,035926
21	0,039779
22	0,049509

Tabela 7 – Probabilidade de cada andar ser o andar de entrada de um passageiro

A segunda tabela contém as probabilidades para cada andar de saída ser o destino de um passageiro, dado seu andar de entrada. Essa tabela pode ser encontrada no Anexo 2.

A partir dessas tabelas, foram montados os códigos, apresentados no item 4.1.2 – Premissa consideradas, que determinam os andares de entrada e saída de cada passageiro.

5. RESULTADOS

5.1. Coleta de dados da simulação

Para o estudo dos resultados da simulação, os dados foram coletados através de arquivos .txt, no qual as informações de tempo são armazenadas por meio da criação de componentes de texto no modelo e gravadas com códigos em linguagem de programação Java.

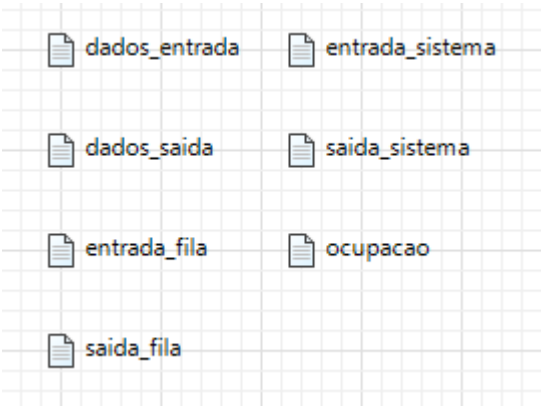


Figura 18 – Componentes .txt para armazenamento dos dados resultados da simulação

O arquivo dados_entrada salva os dados de cada entrada de um passageiro em algum elevador, armazenando as informações de qual elevador está recebendo passageiros, seu andar atual, a quantidade de passageiros que entraram e o tempo em que houveram estas entradas. Analogamente, o arquivo dados_saida armazena as mesmas informações para cada saída de passageiros de determinado elevador.

Os arquivos entrada_fila e saida_fila armazenam as informações dos momentos de entrada e saída, respectivamente, de um passageiro da fila de espera. A partir da diferença entre o tempo de saída e o tempo de entrada, foi possível calcular o tempo de permanência na fila de espera pelos elevadores de cada passageiro.

O arquivo entrada_sistema guarda as informações de chegada de cada passageiro no sistema, o momento de sua chegada, seu andar de entrada e andar de saída. A partir desse arquivo foram criadas as tabelas de validação do modelo, com as probabilidades de cada andar ser a entrada ou a saída de um passageiro. Já o arquivo saida_sistema armazena as informações do tempo de saída do sistema de cada passageiro. Junto com o arquivo entrada_sistema, foi possível calcular o tempo de permanência no sistema de cada passageiro, a partir da diferença do tempo de saída e o tempo de entrada de cada um.

E por fim, o arquivo ocupacao recebe a quantidade de passageiros que determinado elevador está transportando em seu deslocamento e o momento em que este inicia seu movimento.

5.2. Tratamento dos Dados de Saída

Para cada política de decisão, foram coletados os arquivos .txt apresentados no tópico anterior e estes foram exportados para tabelas do Excel onde foram tratados, e a partir dos quais foram elaborados os resultados da simulação.

Dos arquivos entrada_fila e saida_fila, foi montada uma tabela com a diferença entre os tempos de saída e entrada na fila para cada passageiro. De maneira que foi obtido o tempo de permanência na fila de espera para cada um. Analogamente, foram feitos os mesmos cálculos para o tempo de permanência de cada passageiro no sistema, utilizando os arquivos entrada_sistema e saida_sistema.

No arquivo entrada_fila também foi possível extrair os andares de chegada de cada passageiro, assim sendo possível analisar o tamanho das filas de espera pelos elevadores separadas por andar. Esse cálculo foi feito utilizando o Jupyter Notebook para análise dos dados com linguagem de programação Python e considerando intervalos de 60 segundos para evitar repetitividade das informações. A partir desses dados foi encontrado o tamanho médio das filas para cada andar.

Além disso, também foram calculadas as taxas de ocupação média para cada elevador. O arquivo ocupacao guarda as informações do tempo e do número de passageiros que cada elevador contém durante seus deslocamentos, de maneira que foi possível montar uma tabela com o número de passageiros dentro de cada elevador em intervalos de 60 segundos. A partir dessa tabela foram encontradas as taxas de ocupação média de cada elevador, na qual foram consideradas a porcentagem de viagens realizadas por cada elevador com ou sem passageiros. Essa análise também foi feita por meio do Jupyter Notebook.

5.3. Resultados da simulação

Para comparação da eficiência das políticas de decisão foram utilizados os indicadores tempo de espera na fila, tempo de permanência no sistema ou tempo de viagem, tamanho da fila de espera e ocupação dos elevadores. Estes apresentados a seguir:

Política 1 – Atende o andar mais baixo	
Tempo médio de espera na fila (segundos)	28,763254
Tempo médio de permanência no sistema (segundos)	54,285393
Política 2 – Elevador mantém a direção do seu movimento	
Tempo médio de espera na fila (segundos)	27,862421
Tempo médio de permanência no sistema (segundos)	52,466312
Política 3 – Elevador atende o andar mais próximo ao seu andar atual	
Tempo médio de espera na fila (segundos)	27,630957
Tempo médio de permanência no sistema (segundos)	51,838331

Tabela 8 – Resultados da simulação – tempo de espera na fila e tempo de permanência no sistema - para cada política de decisão

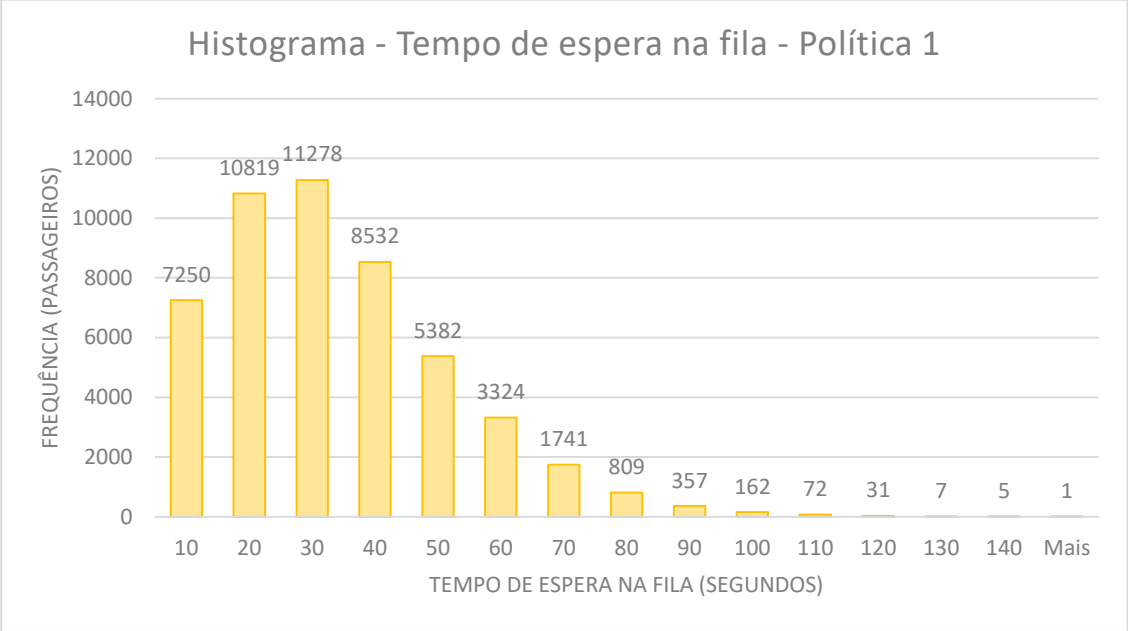


Figura 19 – Histograma do tempo de espera na fila – política 1

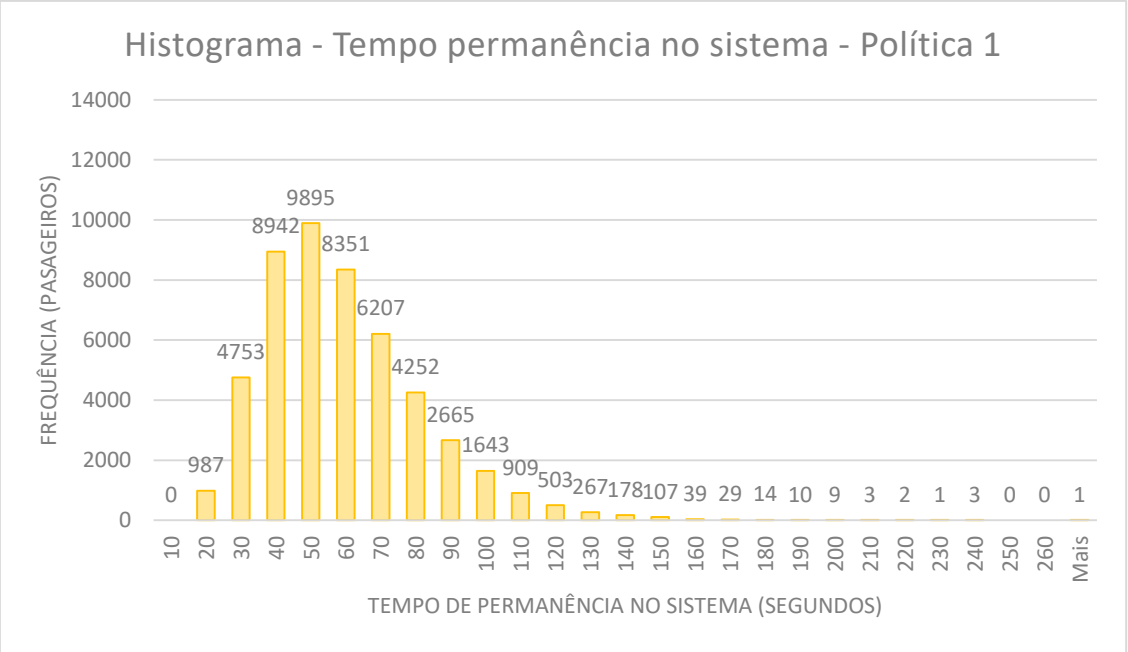


Figura 20 - Histograma do tempo de permanência no sistema – política 1

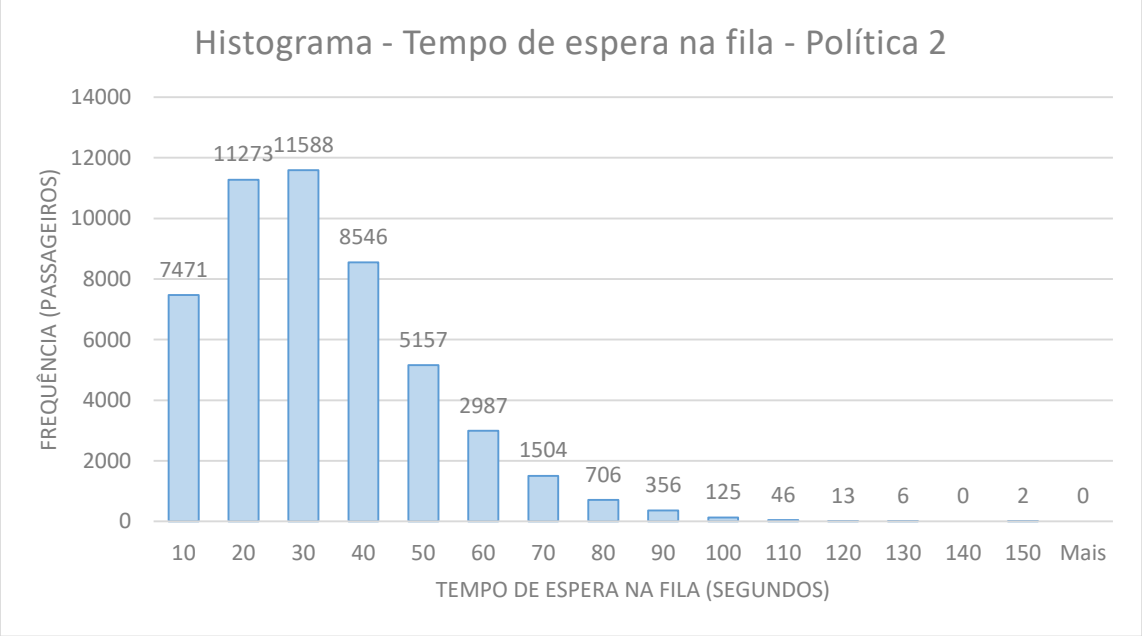


Figura 21 – Histograma do tempo de espera na fila – política 2

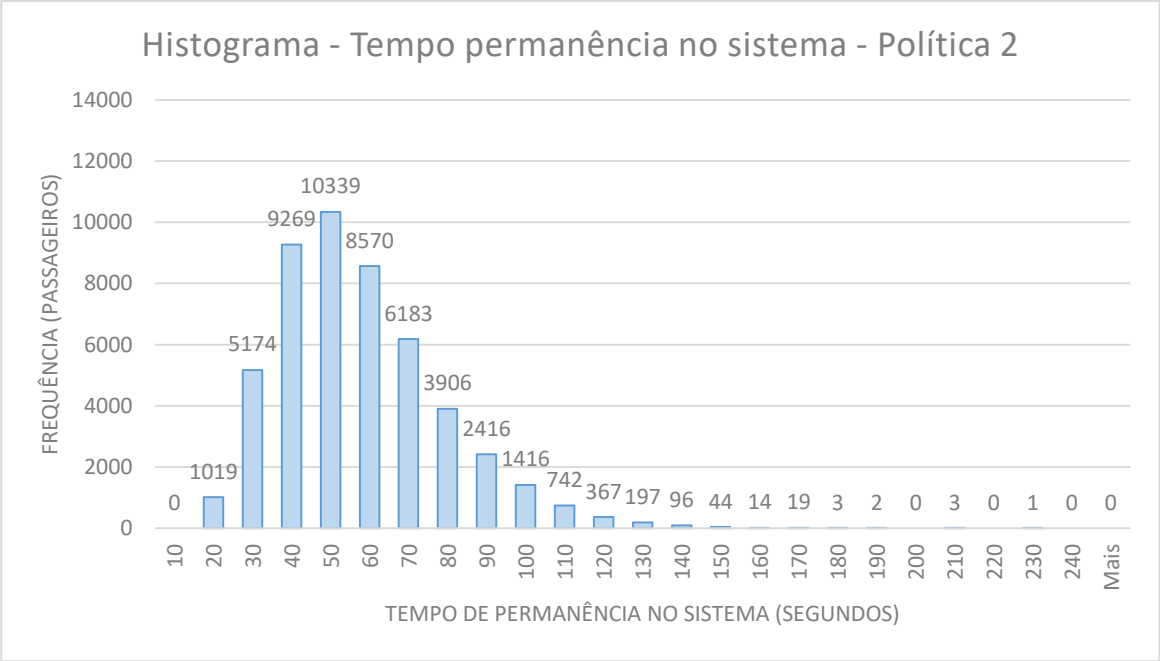


Figura 22 – Histograma do tempo de permanência no sistema – política 2

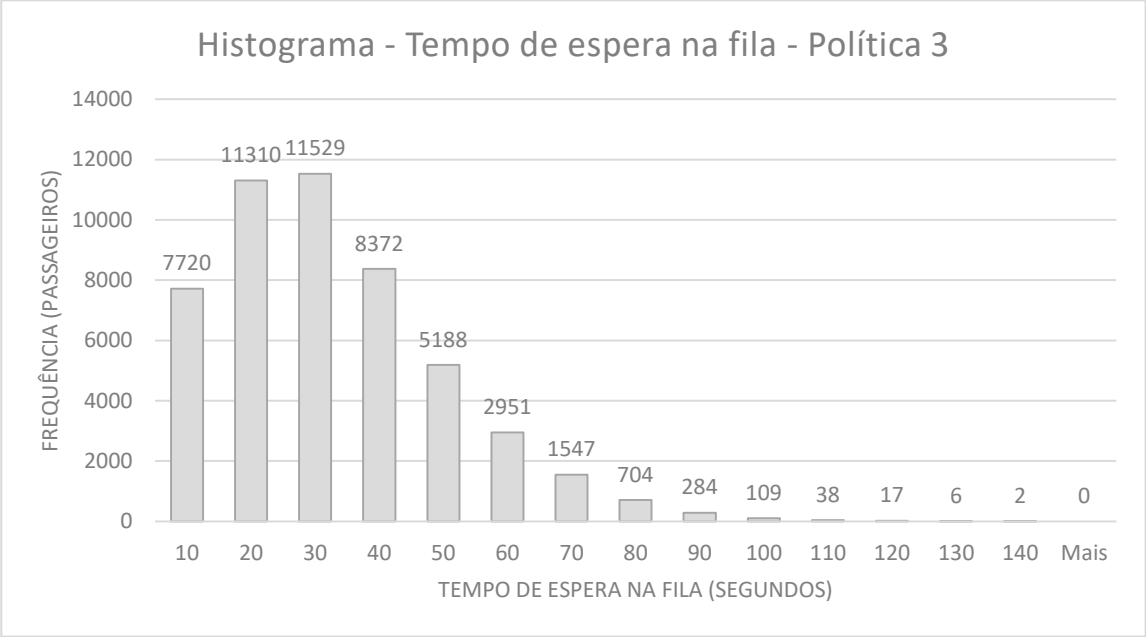


Figura 23 – Histograma do tempo de espera na fila – política 3

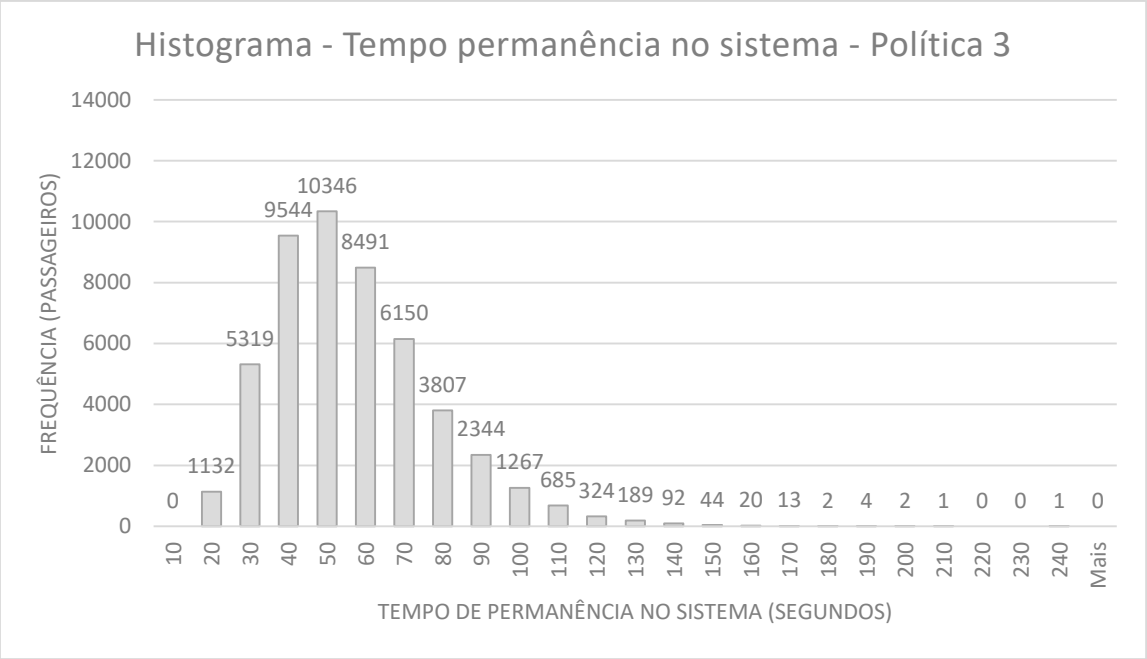


Figura 24 – Histograma do tempo de permanência no sistema – política 3

Taxa de ocupação dos elevadores para cada política de decisão					
Política 1		**Política 2**		**Política 3**	
Elevador	Taxa de ocupação	Elevador	Taxa de ocupação	Elevador	Taxa de ocupação
0	0,512126	0	0,498141	0	0,499026
1	0,501151	1	0,497256	1	0,505576
2	0,508586	2	0,498584	2	0,499911
3	0,507878	3	0,508763	3	0,498495
4	0,510887	4	0,505576	4	0,507878
5	0,500974	5	0,508322	5	0,490175

Tabela 9 – Taxa de ocupação do elevador para cada política de tomada de decisão

Tamanho da fila de espera para cada andar do prédio					
Política 1		**Política 2**		**Política 3**	
Andar	Tamanho da fila médio	Elevador	Tamanho da fila médio	Elevador	Tamanho da fila médio
-4	0,153751	-4	0,138451	-4	0,148023
-3	0,181592	-3	0,173374	-3	0,175918
-2	0,138145	-2	0,138145	-2	0,1252
-1	0,18422	-1	0,167553	-1	0,18106
0	0,14638	0	0,130627	0	0,135178
1	0,143922	1	0,134728	1	0,135638
2	0,168085	2	0,148785	2	0,148936
3	0,191278	3	0,184607	3	0,185882
4	0,146631	4	0,139716	4	0,144301
5	0,098349	5	0,102482	5	0,099113
6	0,139589	6	0,133381	6	0,133251
7	0,10692	7	0,115937	7	0,110678
8	0,103901	8	0,096774	8	0,106383
9	0,108669	9	0,108333	9	0,110106
10	0,122268	10	0,116094	10	0,119808
11	0,137662	11	0,129787	11	0,127505
12	0,111781	12	0,113241	12	0,104628
13	0,152987	13	0,142756	13	0,151746
14	0,022103	14	0,022913	14	0,022396
15	0,159482	15	0,141007	15	0,139514
16	0,149849	16	0,152004	16	0,150532
17	0,147294	17	0,143946	17	0,145922
18	0,149087	18	0,153982	18	0,143085
19	0,145035	19	0,13136	19	0,130473
20	0,151241	20	0,160128	20	0,156605
21	0,161611	21	0,165396	21	0,156588
22	0,189218	22	0,196809	22	0,182414

Tabela 10 – Tamanho da fila de espera para cada andar e para cada política de tomada de decisão

A partir desses dados, observa-se que não há grande variação dos tempos de espera médio e tempos médios de permanência no sistema entre as três políticas de decisão.

Em relação às taxas de ocupação, estas estão próximas a 50%, o que significa que aproximadamente metade das viagens realizadas pelos elevadores ocorrem sem passageiros, representando as viagens em que os elevadores estão se dirigindo ao andar em que houve um *hall call*.

Além disso, percebe-se que o tamanho médio das filas de cada andar é inferior a 1, mostrando que os andares passam a maior parte do tempo sem filas de espera. O que é condizente com os tempos de espera na fila inferiores a 1 minuto.

De maneira geral, esses resultados mostram um cenário em que os passageiros não precisam esperar muito tempo pela chegada dos elevadores e estes conseguem cumprir bem a demanda. Um ponto importante, que traz bastante impacto à simulação, é a velocidade de operação dos elevadores. Neste cenário foi considerada a velocidade de 4 metros por segundo, o que resulta em um tempo de viagem de 1,0875 segundos por andar, mais rápido do que muitos elevadores residenciais. Para analisar o impacto da velocidade dos elevadores nos tempos de espera e tempo de permanência no sistema, foram simulados outros dois cenários, que consideravam velocidades de 0,5 m/s e 1 m/s, para cada uma das políticas de decisão. Os resultados são apresentados na tabela a seguir:

Velocidades	**0,5 m/s**	**1 m/s**	**4 m/s**
Política 1			
Tempo médio de espera na fila (segundos)	2076,940589	184,526410	28,763254
Tempo médio no sistema (segundos)	4820,050643	409,516911	54,285393
Política 2			
Tempo médio de espera na fila (segundos)	201,026364	139,341591	27,862421
Tempo médio no sistema (segundos)	436,200303	245,932215	52,466312
Política 3			
Tempo médio de espera na fila (segundos)	202,044888	138,298104	27,630957
Tempo médio no sistema (segundos)	435,686612	241,339479	51,838331

Tabela 11 – Resultados das simulações considerando diferentes velocidades de movimentação dos elevadores

Analisando os resultados das políticas 2 e 3, observa-se que um aumento de 100% na velocidade do elevador gera um impacto de redução de aproximadamente 30,7% no tempo de espera na fila e uma redução de aproximadamente 43,6% no tempo de permanência no sistema, refletindo o grande impacto que a velocidade tem sobre o sistema de elevadores. A política 1 apresentou um comportamento diverso, no qual o mesmo aumento da velocidade resultou em uma redução perto de 91,1%, mostrando a diferença de impacto causado por uma política que prioriza os andares inferiores e que, portanto, possui grandes filas nos andares mais altos.

6. CONCLUSÃO

A partir dos resultados obtidos da simulação, conclui-se que não há grandes diferenças entre os tempos de espera médio na fila e os tempos médios de permanência no sistema (ou tempos de viagem) para as três políticas de decisão, considerando as premissas informadas pela equipe de segurança do instituto e as informações resultantes da análise da taxa de ocupação dos elevadores e do tamanho das filas de cada andar.

Entretanto, como foi mencionado no tópico de restrições dos dados, o modelo foi montado a partir das taxas de acionamento dos elevadores e não do número de pessoas que realmente entraram ou saíram dos elevadores, perdendo a precisão dos dados. Além disso, não foram consideradas as variações ao longo do dia, não havendo distinção entre os horários de pico (nos quais há maior demanda pelos elevadores e consequentemente maior tempo de espera) e os horários em que o fluxo de pessoas é reduzido. Dessa maneira, o alto tempo de espera pelos elevadores acabou sendo diluído nos momentos em que o sistema de elevadores possui baixa demanda, resultando no baixo tempo de espera, médio, ao contrário do que foi comentado pelos funcionários do ICESP durante a reunião.

Essa diluição da demanda ao longo do dia refletiu, não apenas a redução dos tempos de espera, mas também no tamanho da fila, uma vez que em horários de pico nos quais poderiam chegar 10 pessoas ao mesmo tempo, esse volume acaba sendo distribuído em diferentes momentos, reduzindo o tamanho das filas.

Os resultados também permitiram concluir que a velocidade dos elevadores é um fator de grande impacto sobre o tempo de espera. A velocidade informada pela equipe de segurança do ICESP não levou em consideração os tempos de aceleração e desaceleração dos elevadores que aumentam o tempo de viagem entre dois andares. Dessa forma, pode-se afirmar que a velocidade real dos elevadores é algo entre os valores simulados para 4 m/s e 1 m/s e apresentaria resultados entre os tempos de espera para essas duas velocidades.

Por fim, não é possível afirmar qual política de decisão é a mais eficiente, uma vez que as políticas 2 e 3 apresentaram resultados semelhantes em todos os cenários de velocidades, porém é plausível dizer que quanto menor a velocidade dos elevadores, menos eficiente é a política 1 em relação as outras duas políticas.

REFERÊNCIAS BIBLIOGRÁFICAS

BORSHCHEV, A.; FILIPPOV, A. **From System Dynamics and Discrete Event to Practical Agent Based Modeling: Reasons, Techniques, Tools**. The 22nd International Conference of the System Dynamics Society, July 25 - 29, 2004, Oxford, England.

VELTEN, K. **Mathematical Modeling and Simulation: Introduction for Scientists and Engineers.** 2009.

BONOMO, R. **Metodologia da Pesquisa Científica**. DocPlayer, 2015. Disponível em: https://docplayer.com.br/3904967-Metodologia-da-pesquisa-cientifica-prof-robson-bonomo-baseado-em-ifsc-sbisbi.html.

LIMA, T.; FARIA, S.; SOARES, B.; CARNEIRO, T. **Modelagem de sistemas baseada em agentes: alguns conceitos e ferramentas.** Anais XIV Simpósio Brasileiro de Sensoriamento Remoto, Natal, Brasil, 25-30 abril 2009.

BRITO, T.; BOTTER, R. **Uma comparação conceitual entre as metodologias de simulação discreta e a contínua como elemento impulsionador da simulação híbrida**. Rio de Janeiro, v.6, n.2, p. 202-225, 2014.

ALVES COSTEIRA, E. M. **Arquitetura hospitalar: história, evolução e novas visões**. [s.l: s.n.].

CORTÉS, P.; LARRAÑETA, J.; ONIEVA, L. **Genetic algorithm for controllers in elevator groups: Analysis and simulation during lunchpeak traffic.** Applied Soft Computing Journal, v. 4, n. 2, p. 159–174, 2004.

MUÑOZ, D. M. et al. **Implementation of dispatching algorithms for elevator systems using reconfigurable architectures.** SBCCI 2006 - 19th Symposium on Integrated Circuits and Systems Design, v. 2006, p. 32–37, 2006.

KLÜBER, T. E.; BURAK, D. **Concepções de modelagem matemática: contribuições teóricas**. Educação Matemática Pesquisa: Revista do Programa de Estudos Pós-Graduados em Educação Matemática, v. 10, n. 1, p. 17–34, 2008. Disponível em: <https://revistas.pucsp.br/emp/article/view/1642/1058>.

KOEHLER, J.; OTTIGER, D. **An AI-based approach to destination control in elevators**. AI Magazine, v. 23, n. 3, p. 59–78, 2002.

MUÑOZ ARBOLEDA, D. M. **Implementação e simulação de algoritmos de escalonamento para sistemas de elevadores usando arquiteturas reconfiguráveis**. 2006.

VAN, L. Da et al. **An Intelligent Elevator Development and Management System**. IEEE Systems Journal, v. 14, n. 2, p. 3015–3026, 2020.

KIM, C. B. et al. **Design and implementation of a fuzzy elevator group control system**. IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans., v. 28, n. 3, p. 277–287, 1998.

NUNES HENRIQUES, M. F. **Simulação e melhoria de um sistema de gestão de elevadores**. Universidade do Minho Escola de Engenharia, 2016.

KREUZPOINTNER, G.; MEINDL, X.; TÄUBER, R. **A single chip mos decoder ic for a video program identification system**. IEEE Transactions on Consumer Electronics, v. 34, n. 4, p. 847–855, 1988.

BEADLE, L. et al. **Energy-Efficient Elevators for Tall Buildings.** Tall Buildings and Urban Habitat, n. December, 2001.

BARNEY, G.; AL-SHARIF, L. **Elevator traffic handbook: Theory and practice: Second edition**. [s.l: s.n.]

TASCHETTO MATTOS, G. M.; VANZELLA, P. **Simulação e Plannig aplicados a elevadores**. Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, 2016

CRITES, R. H.; BARTO A.G. **Elevator Group Control Using Multiple Reinforcement Learning Agents.** Machine Learning. Kluwer Academic Publishers, Boston. Manufactured in The Netherlands. 1998

HANGLI, G. et al. **Intellevator: An Intelligent Elevator System Proactive in Traffic Control for Time-Efficiency Improvement.** IEEE Access, 2020.

SHI, D.; XU, B. **Intelligent elevator control and safety monitoring system**. IOP Conference Series: Materials Science and Engineering, v. 366, n. 1, 2018.

LIU, J.; LIU, Y. **Ant colony algorithm and fuzzy neural network-based intelligent dispatching algorithm of an elevator group control system**. 2007 IEEE International Conference on Control and Automation, ICCA, v. 00, p. 2306–2310, 2007.

WAINER, J. **Métodos de pesquisa quantitativa e qualitativa para a Ciência da Computação**. Atualização em Informática. Org: Tomasz Kowaltowski, n. September, p. 1–42, 2007. Disponível em: <http://www.pucrs.br/famat/viali/educem/material/textos/Pesquisa.pdf>.

BORSHCHEV, A.; FILIPPOV, A. **From System Dynamics and Discrete Event to Practical Agent Based Modeling: Reasons, Techniques, Tools.** The 22nd International Conference of the System Dynamics Society, July 25 - 29, 2004, Oxford, England

AUTOR, P. O. et al. **Estudo e Análise de Desempenho de Algoritmos para Elevadores Inteligentes.** Universidade Federal do Rio de Janeiro. 2007.

MUÑOZ, D. M. et al. **Implementation of dispatching algorithms for elevator systems using reconfigurable architectures**. SBCCI 2006 - 19th Symposium on Integrated Circuits and Systems Design, v. 2006, p. 32–37, 2006.

CAMARA, T. et al. **Otimização temporal para de sistemas de elevadores a partir de simulação Monte Carlo.** [s.l: s.n.]

BAPIN, Y.; ZARIKAS, V. **Smart building's elevator with intelligent control algorithm based on Bayesian networks**. International Journal of Advanced Computer Science and Applications, v. 10, n. 2, p. 16–24, 2019.

SIIKONEN, M. L.; SUSI, T.; HAKONEN, H. **Passenger traffic flow simulation in tall buildings**. Elevator World, v. 49, n. 8, p. 117–123, 2001.

SIIKONEN, M. **People Flow and Automated Transportation**. Kone. March, 2005.

RONG, A.; HAKONEN, H.; LAHDELMA, R. **Estimated Time of Arrival (ETA) Based Elevator Group Control Algorithm with More Accurate Estimation**. n. 584, 2003.

ZHANG, X.; FU, A. Y.; ZHANG, J. **Research on Scheduling Algorithm Based on Elevator Group Control System.** v. 136, n. Icadme, p. 237–243, 2017.

CARLO, H. J.; VIS, I. F. A. **Sequencing dynamic storage systems with multiple lifts and shuttles.** International Journal of Production Economics, 2012. .

ANEXO 1 – Código da função que determina o andar de saída dos passageiros

```
Random random = new Random();
double numero = random.nextDouble();
int andarSaida = 0;

switch(andarEntrada) {
	case -4:
	if(numero <= 0.037409) {andarSaida = 1;}
	if(numero > 0.037409 && numero <= 0.115392) {andarSaida = 2;}
	if(numero > 0.115392 && numero <= 0.170110) {andarSaida = 3;}
	if(numero > 0.170110 && numero <= 0.255351) {andarSaida = 4;}
	if(numero > 0.255351 && numero <= 0.287363) {andarSaida = 5;}
	if(numero > 0.287363 && numero <= 0.338917) {andarSaida = 6;}
	if(numero > 0.338917 && numero <= 0.384701) {andarSaida = 7;}
	if(numero > 0.384701 && numero <= 0.400149) {andarSaida = 9;}
	if(numero > 0.400149 && numero <= 0.425647) {andarSaida = 10;}
	if(numero > 0.425647 && numero <= 0.444072) {andarSaida = 11;}
	if(numero > 0.444072 && numero <= 0.483901) {andarSaida = 12;}
	if(numero > 0.483901 && numero <= 0.519821) {andarSaida = 13;}
	if(numero > 0.519821 && numero <= 0.571003) {andarSaida = 15;}
	if(numero > 0.571003 && numero <= 0.576028) {andarSaida = 16;}
	if(numero > 0.576028 && numero <= 0.636888) {andarSaida = 17;}
	if(numero > 0.636888 && numero <= 0.673925) {andarSaida = 18;}
	if(numero > 0.673925 && numero <= 0.684348) {andarSaida = 19;}
	if(numero > 0.684348 && numero <= 0.756561) {andarSaida = 20;}
	if(numero > 0.756561 && numero <= 0.813140) {andarSaida = 22;}
	if(numero > 0.813140 && numero <= 0.893728) {andarSaida = -1;}
	if(numero > 0.893728 && numero <= 0.903406) {andarSaida = -2;}
	if(numero > 0.903406 && numero <= 0.920156) {andarSaida = -3;}
	if(numero > 0.920156 && numero <= 1) {andarSaida = 0;}
	break;

	case -3:
	if(numero <= 0.075923) {andarSaida = 1;}
	if(numero > 0.075923 && numero <= 0.098993) {andarSaida = 2;}
	if(numero > 0.098993 && numero <= 0.138563) {andarSaida = 3;}
	if(numero > 0.138563 && numero <= 0.178551) {andarSaida = 4;}
	if(numero > 0.178551 && numero <= 0.211689) {andarSaida = 5;}
	if(numero > 0.211689 && numero <= 0.245666) {andarSaida = 6;}
	if(numero > 0.245666 && numero <= 0.302013) {andarSaida = 7;}
	if(numero > 0.302013 && numero <= 0.325503) {andarSaida = 8;}
	if(numero > 0.325503 && numero <= 0.379195) {andarSaida = 9;}
	if(numero > 0.379195 && numero <= 0.427852) {andarSaida = 10;}
	if(numero > 0.427852 && numero <= 0.480145) {andarSaida = 11;}
	if(numero > 0.480145 && numero <= 0.529502) {andarSaida = 12;}
	if(numero > 0.529502 && numero <= 0.569491) {andarSaida = 13;}
	if(numero > 0.569502 && numero <= 0.578300) {andarSaida = 14;}
	if(numero > 0.578300 && numero <= 0.622903) {andarSaida = 15;}
	if(numero > 0.622903 && numero <= 0.672260) {andarSaida = 16;}
	if(numero > 0.672260 && numero <= 0.714066) {andarSaida = 17;}
	if(numero > 0.714066 && numero <= 0.756572) {andarSaida = 18;}
	if(numero > 0.756572 && numero <= 0.792925) {andarSaida = 19;}
	if(numero > 0.792925 && numero <= 0.808026) {andarSaida = 20;}
	if(numero > 0.808026 && numero <= 0.824105) {andarSaida = 21;}
	if(numero > 0.824105 && numero <= 0.878356) {andarSaida = 22;}
	if(numero > 0.878356 && numero <= 0.948266) {andarSaida = -1;}
	if(numero > 0.948266 && numero <= 0.953440) {andarSaida = -2;}
	if(numero > 0.953440 && numero <= 1) {andarSaida = 0;}
```

```
break;

case -2:
if(numero <= 0.036412) {andarSaida = 1;}
if(numero > 0.036412 && numero <= 0.051261) {andarSaida = 2;}
if(numero > 0.051261 && numero <= 0.112490) {andarSaida = 3;}
if(numero > 0.112490 && numero <= 0.128153) {andarSaida = 4;}
if(numero > 0.128153 && numero <= 0.154394) {andarSaida = 5;}
if(numero > 0.154394 && numero <= 0.205045) {andarSaida = 6;}
if(numero > 0.205045 && numero <= 0.287225) {andarSaida = 7;}
if(numero > 0.287225 && numero <= 0.311229) {andarSaida = 9;}
if(numero > 0.311229 && numero <= 0.340928) {andarSaida = 10;}
if(numero > 0.340928 && numero <= 0.379984) {andarSaida = 11;}
if(numero > 0.379984 && numero <= 0.439788) {andarSaida = 12;}
if(numero > 0.439788 && numero <= 0.498373) {andarSaida = 13;}
if(numero > 0.498373 && numero <= 0.517087) {andarSaida = 14;}
if(numero > 0.517087 && numero <= 0.532140) {andarSaida = 15;}
if(numero > 0.532140 && numero <= 0.599064) {andarSaida = 16;}
if(numero > 0.599064 && numero <= 0.671481) {andarSaida = 17;}
if(numero > 0.671481 && numero <= 0.713995) {andarSaida = 18;}
if(numero > 0.713995 && numero <= 0.763832) {andarSaida = 19;}
if(numero > 0.763832 && numero <= 0.787632) {andarSaida = 20;}
if(numero > 0.787632 && numero <= 0.842758) {andarSaida = 21;}
if(numero > 0.842758 && numero <= 0.916599) {andarSaida = 22;}
if(numero > 0.916599 && numero <= 0.940195) {andarSaida = -1;}
if(numero > 0.940195 && numero <= 0.943247) {andarSaida = -3;}
if(numero > 0.943247 && numero <= 0.945688) {andarSaida = -4;}
if(numero > 0.945688 && numero <= 1) {andarSaida = 0;}
break;

case -1:
if(numero <= 0.062437) {andarSaida = 1;}
if(numero > 0.062437 && numero <= 0.107305) {andarSaida = 2;}
if(numero > 0.107305 && numero <= 0.173660) {andarSaida = 3;}
if(numero > 0.173660 && numero <= 0.219414) {andarSaida = 4;}
if(numero > 0.219414 && numero <= 0.269843) {andarSaida = 5;}
if(numero > 0.269843 && numero <= 0.309783) {andarSaida = 6;}
if(numero > 0.309783 && numero <= 0.363625) {andarSaida = 7;}
if(numero > 0.363625 && numero <= 0.396992) {andarSaida = 8;}
if(numero > 0.396992 && numero <= 0.434277) {andarSaida = 9;}
if(numero > 0.434277 && numero <= 0.472194) {andarSaida = 10;}
if(numero > 0.472194 && numero <= 0.514788) {andarSaida = 11;}
if(numero > 0.514788 && numero <= 0.548534) {andarSaida = 12;}
if(numero > 0.548534 && numero <= 0.590875) {andarSaida = 13;}
if(numero > 0.590875 && numero <= 0.601997) {andarSaida = 14;}
if(numero > 0.601997 && numero <= 0.640167) {andarSaida = 15;}
if(numero > 0.640167 && numero <= 0.667467) {andarSaida = 16;}
if(numero > 0.667467 && numero <= 0.706395) {andarSaida = 17;}
if(numero > 0.706395 && numero <= 0.740268) {andarSaida = 18;}
if(numero > 0.740268 && numero <= 0.778185) {andarSaida = 19;}
if(numero > 0.778185 && numero <= 0.811805) {andarSaida = 20;}
if(numero > 0.811805 && numero <= 0.847826) {andarSaida = 21;}
if(numero > 0.847826 && numero <= 0.879803) {andarSaida = 22;}
if(numero > 0.879803 && numero <= 0.885490) {andarSaida = -2;}
if(numero > 0.885490 && numero <= 0.935667) {andarSaida = -3;}
if(numero > 0.935667 && numero <= 0.973837) {andarSaida = -4;}
if(numero > 0.973837 && numero <= 1) {andarSaida = 0;}
break;
```

```
case 0:
if(numero <= 0.028280) {andarSaida = 1;}
if(numero > 0.028280 && numero <= 0.063707) {andarSaida = 2;}
if(numero > 0.063707 && numero <= 0.118139) {andarSaida = 3;}
if(numero > 0.118139 && numero <= 0.183214) {andarSaida = 4;}
if(numero > 0.183214 && numero <= 0.236430) {andarSaida = 5;}
if(numero > 0.236430 && numero <= 0.283564) {andarSaida = 6;}
if(numero > 0.283564 && numero <= 0.330698) {andarSaida = 7;}
if(numero > 0.330698 && numero <= 0.370382) {andarSaida = 8;}
if(numero > 0.370382 && numero <= 0.395925) {andarSaida = 9;}
if(numero > 0.395925 && numero <= 0.437281) {andarSaida = 10;}
if(numero > 0.437281 && numero <= 0.475901) {andarSaida = 11;}
if(numero > 0.475901 && numero <= 0.507526) {andarSaida = 12;}
if(numero > 0.507526 && numero <= 0.558157) {andarSaida = 13;}
if(numero > 0.558157 && numero <= 0.596016) {andarSaida = 15;}
if(numero > 0.596016 && numero <= 0.648016) {andarSaida = 16;}
if(numero > 0.648016 && numero <= 0.701384) {andarSaida = 17;}
if(numero > 0.701384 && numero <= 0.733009) {andarSaida = 18;}
if(numero > 0.733009 && numero <= 0.785160) {andarSaida = 19;}
if(numero > 0.785160 && numero <= 0.834119) {andarSaida = 20;}
if(numero > 0.834119 && numero <= 0.861487) {andarSaida = 21;}
if(numero > 0.861487 && numero <= 0.920480) {andarSaida = 22;}
if(numero > 0.920480 && numero <= 0.935229) {andarSaida = -1;}
if(numero > 0.935229 && numero <= 0.965182) {andarSaida = -3;}
if(numero > 0.965182 && numero <= 1) {andarSaida = -4;}
break;

case 1:
if(numero <= 0.036632) {andarSaida = 2;}
if(numero > 0.036632 && numero <= 0.063021) {andarSaida = 3;}
if(numero > 0.063021 && numero <= 0.087674) {andarSaida = 4;}
if(numero > 0.087674 && numero <= 0.151736) {andarSaida = 5;}
if(numero > 0.151736 && numero <= 0.207813) {andarSaida = 6;}
if(numero > 0.207813 && numero <= 0.224479) {andarSaida = 7;}
if(numero > 0.224479 && numero <= 0.233681) {andarSaida = 8;}
if(numero > 0.233681 && numero <= 0.299653) {andarSaida = 10;}
if(numero > 0.299653 && numero <= 0.357292) {andarSaida = 11;}
if(numero > 0.357292 && numero <= 0.425868) {andarSaida = 12;}
if(numero > 0.425868 && numero <= 0.495313) {andarSaida = 13;}
if(numero > 0.495313 && numero <= 0.515799) {andarSaida = 14;}
if(numero > 0.515799 && numero <= 0.587847) {andarSaida = 15;}
if(numero > 0.587847 && numero <= 0.616146) {andarSaida = 16;}
if(numero > 0.616146 && numero <= 0.623785) {andarSaida = 17;}
if(numero > 0.623785 && numero <= 0.696528) {andarSaida = 18;}
if(numero > 0.696528 && numero <= 0.736632) {andarSaida = 19;}
if(numero > 0.736632 && numero <= 0.786458) {andarSaida = 21;}
if(numero > 0.786458 && numero <= 0.845660) {andarSaida = 22;}
if(numero > 0.845660 && numero <= 0.850521) {andarSaida = -1;}
if(numero > 0.850521 && numero <= 0.907639) {andarSaida = -2;}
if(numero > 0.907639 && numero <= 0.943750) {andarSaida = -3;}
if(numero > 0.943750 && numero <= 0.975868) {andarSaida = -4;}
if(numero > 0.975868 && numero <= 1) {andarSaida = 0;}
break;

case 2:
if(numero <= 0.016678) {andarSaida = 1;}
if(numero > 0.016678 && numero <= 0.051534) {andarSaida = 3;}
if(numero > 0.051534 && numero <= 0.062875) {andarSaida = 4;}
if(numero > 0.062875 && numero <= 0.090560) {andarSaida = 5;}
```

```
if(numero > 0.090560 && numero <= 0.135924) {andarSaida = 6;}
if(numero > 0.135924 && numero <= 0.181288) {andarSaida = 7;}
if(numero > 0.181288 && numero <= 0.183456) {andarSaida = 8;}
if(numero > 0.183456 && numero <= 0.265344) {andarSaida = 9;}
if(numero > 0.265344 && numero <= 0.337225) {andarSaida = 10;}
if(numero > 0.337225 && numero <= 0.393429) {andarSaida = 11;}
if(numero > 0.393429 && numero <= 0.462308) {andarSaida = 12;}
if(numero > 0.462308 && numero <= 0.517845) {andarSaida = 13;}
if(numero > 0.517845 && numero <= 0.529853) {andarSaida = 14;}
if(numero > 0.529853 && numero <= 0.592395) {andarSaida = 15;}
if(numero > 0.592395 && numero <= 0.595730) {andarSaida = 16;}
if(numero > 0.595730 && numero <= 0.622915) {andarSaida = 17;}
if(numero > 0.622915 && numero <= 0.669613) {andarSaida = 18;}
if(numero > 0.669613 && numero <= 0.694630) {andarSaida = 19;}
if(numero > 0.694630 && numero <= 0.710307) {andarSaida = 20;}
if(numero > 0.710307 && numero <= 0.761508) {andarSaida = 21;}
if(numero > 0.761508 && numero <= 0.793195) {andarSaida = 22;}
if(numero > 0.793195 && numero <= 0.879920) {andarSaida = -1;}
if(numero > 0.879920 && numero <= 0.894263) {andarSaida = -2;}
if(numero > 0.894263 && numero <= 0.939960) {andarSaida = -3;}
if(numero > 0.939960 && numero <= 0.983322) {andarSaida = -4;}
if(numero > 0.983322 && numero <= 1) {andarSaida = 0;}
break;

case 3:
if(numero <= 0.039845) {andarSaida = 1;}
if(numero > 0.039845 && numero <= 0.066331) {andarSaida = 2;}
if(numero > 0.066331 && numero <= 0.083089) {andarSaida = 4;}
if(numero > 0.083089 && numero <= 0.116958) {andarSaida = 5;}
if(numero > 0.116958 && numero <= 0.162311) {andarSaida = 6;}
if(numero > 0.162311 && numero <= 0.220555) {andarSaida = 7;}
if(numero > 0.220555 && numero <= 0.277394) {andarSaida = 8;}
if(numero > 0.277394 && numero <= 0.331654) {andarSaida = 9;}
if(numero > 0.331654 && numero <= 0.372788) {andarSaida = 10;}
if(numero > 0.372788 && numero <= 0.428103) {andarSaida = 11;}
if(numero > 0.428103 && numero <= 0.479433) {andarSaida = 12;}
if(numero > 0.479433 && numero <= 0.509786) {andarSaida = 13;}
if(numero > 0.509786 && numero <= 0.560998) {andarSaida = 15;}
if(numero > 0.560998 && numero <= 0.589242) {andarSaida = 16;}
if(numero > 0.589242 && numero <= 0.627681) {andarSaida = 17;}
if(numero > 0.627681 && numero <= 0.673268) {andarSaida = 18;}
if(numero > 0.673268 && numero <= 0.710184) {andarSaida = 19;}
if(numero > 0.710184 && numero <= 0.742060) {andarSaida = 20;}
if(numero > 0.742060 && numero <= 0.784953) {andarSaida = 21;}
if(numero > 0.784953 && numero <= 0.825501) {andarSaida = 22;}
if(numero > 0.825501 && numero <= 0.861479) {andarSaida = -1;}
if(numero > 0.861479 && numero <= 0.881753) {andarSaida = -2;}
if(numero > 0.881753 && numero <= 0.915973) {andarSaida = -3;}
if(numero > 0.915973 && numero <= 0.956991) {andarSaida = -4;}
if(numero > 0.956991 && numero <= 1) {andarSaida = 0;}
break;

case 4:
if(numero <= 0.062370) {andarSaida = 1;}
if(numero > 0.062370 && numero <= 0.114518) {andarSaida = 3;}
if(numero > 0.114518 && numero <= 0.145703) {andarSaida = 6;}
if(numero > 0.145703 && numero <= 0.167186) {andarSaida = 8;}
if(numero > 0.167186 && numero <= 0.216216) {andarSaida = 9;}
if(numero > 0.216216 && numero <= 0.257970) {andarSaida = 10;}
```

```
if(numero > 0.257970 && numero <= 0.292620) {andarSaida = 11;}
if(numero > 0.292620 && numero <= 0.360707) {andarSaida = 12;}
if(numero > 0.360707 && numero <= 0.416147) {andarSaida = 13;}
if(numero > 0.416147 && numero <= 0.441441) {andarSaida = 15;}
if(numero > 0.441441 && numero <= 0.496708) {andarSaida = 16;}
if(numero > 0.496708 && numero <= 0.526507) {andarSaida = 17;}
if(numero > 0.526507 && numero <= 0.570002) {andarSaida = 18;}
if(numero > 0.570002 && numero <= 0.591476) {andarSaida = 19;}
if(numero > 0.591476 && numero <= 0.659910) {andarSaida = 20;}
if(numero > 0.659910 && numero <= 0.686244) {andarSaida = 21;}
if(numero > 0.686244 && numero <= 0.737179) {andarSaida = 22;}
if(numero > 0.737179 && numero <= 0.818607) {andarSaida = -1;}
if(numero > 0.818607 && numero <= 0.829349) {andarSaida = -2;}
if(numero > 0.829349 && numero <= 0.848615) {andarSaida = -3;}
if(numero > 0.848615 && numero <= 0.969681) {andarSaida = -4;}
if(numero > 0.969681 && numero <= 1) {andarSaida = 0;}
break;

case 5:
if(numero <= 0.008220) {andarSaida = 1;}
if(numero > 0.008220 && numero <= 0.014642) {andarSaida = 4;}
if(numero > 0.014642 && numero <= 0.018752) {andarSaida = 7;}
if(numero > 0.018752 && numero <= 0.048549) {andarSaida = 9;}
if(numero > 0.048549 && numero <= 0.131518) {andarSaida = 11;}
if(numero > 0.131518 && numero <= 0.199075) {andarSaida = 12;}
if(numero > 0.199075 && numero <= 0.310557) {andarSaida = 13;}
if(numero > 0.310557 && numero <= 0.328795) {andarSaida = 15;}
if(numero > 0.328795 && numero <= 0.422040) {andarSaida = 16;}
if(numero > 0.422040 && numero <= 0.455176) {andarSaida = 17;}
if(numero > 0.455176 && numero <= 0.556126) {andarSaida = 18;}
if(numero > 0.556126 && numero <= 0.589520) {andarSaida = 19;}
if(numero > 0.589520 && numero <= 0.640894) {andarSaida = 20;}
if(numero > 0.640894 && numero <= 0.654508) {andarSaida = 21;}
if(numero > 0.654508 && numero <= 0.741331) {andarSaida = 22;}
if(numero > 0.741331 && numero <= 0.816594) {andarSaida = -1;}
if(numero > 0.816594 && numero <= 0.850758) {andarSaida = -2;}
if(numero > 0.850758 && numero <= 0.930131) {andarSaida = -3;}
if(numero > 0.930131 && numero <= 0.933213) {andarSaida = -4;}
if(numero > 0.933213 && numero <= 1) {andarSaida = 0;}
break;

case 6:
if(numero <= 0.052329) {andarSaida = 1;}
if(numero > 0.052329 && numero <= 0.055298) {andarSaida = 2;}
if(numero > 0.055298 && numero <= 0.157543) {andarSaida = 3;}
if(numero > 0.157543 && numero <= 0.174058) {andarSaida = 8;}
if(numero > 0.174058 && numero <= 0.224717) {andarSaida = 9;}
if(numero > 0.224717 && numero <= 0.244943) {andarSaida = 10;}
if(numero > 0.244943 && numero <= 0.261087) {andarSaida = 11;}
if(numero > 0.261087 && numero <= 0.348117) {andarSaida = 12;}
if(numero > 0.348117 && numero <= 0.369271) {andarSaida = 13;}
if(numero > 0.369271 && numero <= 0.383745) {andarSaida = 14;}
if(numero > 0.383745 && numero <= 0.419558) {andarSaida = 15;}
if(numero > 0.419558 && numero <= 0.474671) {andarSaida = 16;}
if(numero > 0.474671 && numero <= 0.490258) {andarSaida = 17;}
if(numero > 0.490258 && numero <= 0.539061) {andarSaida = 18;}
if(numero > 0.539061 && numero <= 0.560401) {andarSaida = 19;}
if(numero > 0.560401 && numero <= 0.599369) {andarSaida = 20;}
if(numero > 0.599369 && numero <= 0.655595) {andarSaida = 21;}
```

```
if(numero > 0.655595 && numero <= 0.702171) {andarSaida = 22;}
if(numero > 0.702171 && numero <= 0.785860) {andarSaida = -1;}
if(numero > 0.785860 && numero <= 0.827797) {andarSaida = -2;}
if(numero > 0.827797 && numero <= 0.879013) {andarSaida = -3;}
if(numero > 0.879013 && numero <= 0.937280) {andarSaida = -4;}
if(numero > 0.937280 && numero <= 1) {andarSaida = 0;}
break;

case 7:
if(numero <= 0.057091) {andarSaida = 1;}
if(numero > 0.057091 && numero <= 0.072391) {andarSaida = 2;}
if(numero > 0.072391 && numero <= 0.140671) {andarSaida = 3;}
if(numero > 0.140671 && numero <= 0.182005) {andarSaida = 9;}
if(numero > 0.182005 && numero <= 0.283398) {andarSaida = 10;}
if(numero > 0.283398 && numero <= 0.386846) {andarSaida = 13;}
if(numero > 0.386846 && numero <= 0.469970) {andarSaida = 15;}
if(numero > 0.469970 && numero <= 0.521352) {andarSaida = 17;}
if(numero > 0.521352 && numero <= 0.527974) {andarSaida = 18;}
if(numero > 0.527974 && numero <= 0.532085) {andarSaida = 20;}
if(numero > 0.532085 && numero <= 0.571592) {andarSaida = 21;}
if(numero > 0.571592 && numero <= 0.638502) {andarSaida = 22;}
if(numero > 0.638502 && numero <= 0.723681) {andarSaida = -1;}
if(numero > 0.723681 && numero <= 0.775976) {andarSaida = -2;}
if(numero > 0.775976 && numero <= 0.867778) {andarSaida = -3;}
if(numero > 0.867778 && numero <= 0.920073) {andarSaida = -4;}
if(numero > 0.920073 && numero <= 1) {andarSaida = 0;}
break;

case 8:
if(numero <= 0.101884) {andarSaida = 1;}
if(numero > 0.101884 && numero <= 0.202626) {andarSaida = 3;}
if(numero > 0.202626 && numero <= 0.354452) {andarSaida = 12;}
if(numero > 0.354452 && numero <= 0.477454) {andarSaida = 13;}
if(numero > 0.477454 && numero <= 0.524829) {andarSaida = 17;}
if(numero > 0.524829 && numero <= 0.544806) {andarSaida = 21;}
if(numero > 0.544806 && numero <= 0.628139) {andarSaida = 22;}
if(numero > 0.628139 && numero <= 0.709760) {andarSaida = -1;}
if(numero > 0.709760 && numero <= 0.787671) {andarSaida = -2;}
if(numero > 0.787671 && numero <= 0.815925) {andarSaida = -3;}
if(numero > 0.815925 && numero <= 0.869863) {andarSaida = -4;}
if(numero > 0.869863 && numero <= 1) {andarSaida = 0;}
break;

case 9:
if(numero <= 0.056664) {andarSaida = 1;}
if(numero > 0.056664 && numero <= 0.105937) {andarSaida = 2;}
if(numero > 0.105937 && numero <= 0.195615) {andarSaida = 3;}
if(numero > 0.195615 && numero <= 0.207194) {andarSaida = 6;}
if(numero > 0.207194 && numero <= 0.258684) {andarSaida = 7;}
if(numero > 0.258684 && numero <= 0.287756) {andarSaida = 10;}
if(numero > 0.287756 && numero <= 0.298349) {andarSaida = 12;}
if(numero > 0.298349 && numero <= 0.328652) {andarSaida = 15;}
if(numero > 0.328652 && numero <= 0.367332) {andarSaida = 16;}
if(numero > 0.367332 && numero <= 0.389505) {andarSaida = 17;}
if(numero > 0.389505 && numero <= 0.458241) {andarSaida = 18;}
if(numero > 0.458241 && numero <= 0.516630) {andarSaida = 19;}
if(numero > 0.516630 && numero <= 0.540527) {andarSaida = 21;}
if(numero > 0.540527 && numero <= 0.599409) {andarSaida = 22;}
if(numero > 0.599409 && numero <= 0.694999) {andarSaida = -1;}
```

```
if(numero > 0.694999 && numero <= 0.775314) {andarSaida = -2;}
if(numero > 0.775314 && numero <= 0.850949) {andarSaida = -3;}
if(numero > 0.850949 && numero <= 0.940133) {andarSaida = -4;}
if(numero > 0.940133 && numero <= 1) {andarSaida = 0;}
break;

case 10:
if(numero <= 0.069509) {andarSaida = 1;}
if(numero > 0.069509 && numero <= 0.073062) {andarSaida = 2;}
if(numero > 0.073062 && numero <= 0.130802) {andarSaida = 3;}
if(numero > 0.130802 && numero <= 0.157006) {andarSaida = 4;}
if(numero > 0.157006 && numero <= 0.264712) {andarSaida = 6;}
if(numero > 0.264712 && numero <= 0.276926) {andarSaida = 7;}
if(numero > 0.276926 && numero <= 0.311792) {andarSaida = 9;}
if(numero > 0.311792 && numero <= 0.317788) {andarSaida = 11;}
if(numero > 0.317788 && numero <= 0.355763) {andarSaida = 13;}
if(numero > 0.355763 && numero <= 0.393071) {andarSaida = 15;}
if(numero > 0.393071 && numero <= 0.451033) {andarSaida = 17;}
if(numero > 0.451033 && numero <= 0.537419) {andarSaida = 18;}
if(numero > 0.537419 && numero <= 0.600266) {andarSaida = 19;}
if(numero > 0.600266 && numero <= 0.684655) {andarSaida = 20;}
if(numero > 0.684655 && numero <= 0.716189) {andarSaida = 21;}
if(numero > 0.716189 && numero <= 0.787475) {andarSaida = 22;}
if(numero > 0.787475 && numero <= 0.872974) {andarSaida = -1;}
if(numero > 0.872974 && numero <= 0.948257) {andarSaida = -3;}
if(numero > 0.948257 && numero <= 1) {andarSaida = 0;}
break;

case 11:
if(numero <= 0.045821) {andarSaida = 1;}
if(numero > 0.045821 && numero <= 0.049951) {andarSaida = 2;}
if(numero > 0.049951 && numero <= 0.111898) {andarSaida = 3;}
if(numero > 0.111898 && numero <= 0.134317) {andarSaida = 4;}
if(numero > 0.134317 && numero <= 0.205113) {andarSaida = 5;}
if(numero > 0.205113 && numero <= 0.228909) {andarSaida = 6;}
if(numero > 0.228909 && numero <= 0.249754) {andarSaida = 7;}
if(numero > 0.249754 && numero <= 0.269813) {andarSaida = 9;}
if(numero > 0.269813 && numero <= 0.286529) {andarSaida = 12;}
if(numero > 0.286529 && numero <= 0.294002) {andarSaida = 13;}
if(numero > 0.294002 && numero <= 0.343166) {andarSaida = 15;}
if(numero > 0.343166 && numero <= 0.438348) {andarSaida = 16;}
if(numero > 0.438348 && numero <= 0.497935) {andarSaida = 17;}
if(numero > 0.497935 && numero <= 0.514651) {andarSaida = 18;}
if(numero > 0.514651 && numero <= 0.574041) {andarSaida = 19;}
if(numero > 0.574041 && numero <= 0.647198) {andarSaida = 20;}
if(numero > 0.647198 && numero <= 0.723107) {andarSaida = 21;}
if(numero > 0.723107 && numero <= 0.786037) {andarSaida = 22;}
if(numero > 0.786037 && numero <= 0.856441) {andarSaida = -1;}
if(numero > 0.856441 && numero <= 0.911111) {andarSaida = -2;}
if(numero > 0.911111 && numero <= 0.961455) {andarSaida = -3;}
if(numero > 0.961455 && numero <= 1) {andarSaida = 0;}
break;

case 12:
if(numero <= 0.089683) {andarSaida = 1;}
if(numero > 0.089683 && numero <= 0.133650) {andarSaida = 2;}
if(numero > 0.133650 && numero <= 0.248564) {andarSaida = 3;}
if(numero > 0.248564 && numero <= 0.318261) {andarSaida = 4;}
if(numero > 0.318261 && numero <= 0.396702) {andarSaida = 5;}
```

```
if(numero > 0.396702 && numero <= 0.438171) {andarSaida = 6;}
if(numero > 0.438171 && numero <= 0.457657) {andarSaida = 7;}
if(numero > 0.457657 && numero <= 0.475893) {andarSaida = 11;}
if(numero > 0.475893 && numero <= 0.520360) {andarSaida = 15;}
if(numero > 0.520360 && numero <= 0.539345) {andarSaida = 16;}
if(numero > 0.539345 && numero <= 0.547589) {andarSaida = 17;}
if(numero > 0.547589 && numero <= 0.633525) {andarSaida = 21;}
if(numero > 0.633525 && numero <= 0.671496) {andarSaida = 22;}
if(numero > 0.671496 && numero <= 0.763927) {andarSaida = -1;}
if(numero > 0.763927 && numero <= 0.771671) {andarSaida = -2;}
if(numero > 0.771671 && numero <= 0.876343) {andarSaida = -3;}
if(numero > 0.876343 && numero <= 0.927554) {andarSaida = -4;}

if(numero > 0.927554 && numero <= 1) {andarSaida = 0;}
break;

case 13:
if(numero <= 0.054845) {andarSaida = 1;}
if(numero > 0.054845 && numero <= 0.074290) {andarSaida = 2;}
if(numero > 0.074290 && numero <= 0.145754) {andarSaida = 3;}
if(numero > 0.145754 && numero <= 0.196776) {andarSaida = 4;}
if(numero > 0.196776 && numero <= 0.247964) {andarSaida = 5;}
if(numero > 0.247964 && numero <= 0.328569) {andarSaida = 6;}
if(numero > 0.328569 && numero <= 0.385076) {andarSaida = 7;}
if(numero > 0.385076 && numero <= 0.457038) {andarSaida = 8;}
if(numero > 0.457038 && numero <= 0.525012) {andarSaida = 9;}
if(numero > 0.525012 && numero <= 0.535981) {andarSaida = 10;}
if(numero > 0.535981 && numero <= 0.540469) {andarSaida = 14;}
if(numero > 0.540469 && numero <= 0.570052) {andarSaida = 15;}
if(numero > 0.570052 && numero <= 0.612099) {andarSaida = 17;}
if(numero > 0.612099 && numero <= 0.647997) {andarSaida = 18;}
if(numero > 0.647997 && numero <= 0.674921) {andarSaida = 19;}
if(numero > 0.674921 && numero <= 0.703673) {andarSaida = 21;}
if(numero > 0.703673 && numero <= 0.742895) {andarSaida = 22;}
if(numero > 0.742895 && numero <= 0.811035) {andarSaida = -1;}
if(numero > 0.811035 && numero <= 0.856739) {andarSaida = -2;}
if(numero > 0.856739 && numero <= 0.939006) {andarSaida = -3;}
if(numero > 0.939006 && numero <= 0.955792) {andarSaida = -4;}

if(numero > 0.955792 && numero <= 1) {andarSaida = 0;}
break;

case 14:
if(numero <= 0.121311) {andarSaida = 6;}
if(numero > 0.121311 && numero <= 0.144262) {andarSaida = 17;}
if(numero > 0.144262 && numero <= 0.419672) {andarSaida = 18;}
if(numero > 0.419672 && numero <= 0.491803) {andarSaida = -1;}
if(numero > 0.491803 && numero <= 0.767213) {andarSaida = -3;}
if(numero > 0.767213 && numero <= 1) {andarSaida = 0;}
break;

case 15:
if(numero <= 0.015440) {andarSaida = 1;}
if(numero > 0.015440 && numero <= 0.058369) {andarSaida = 2;}
if(numero > 0.058369 && numero <= 0.136886) {andarSaida = 3;}
if(numero > 0.136886 && numero <= 0.177933) {andarSaida = 4;}
if(numero > 0.177933 && numero <= 0.247034) {andarSaida = 6;}
if(numero > 0.247034 && numero <= 0.252495) {andarSaida = 7;}
if(numero > 0.252495 && numero <= 0.347957) {andarSaida = 10;}
```

```
if(numero > 0.347957 && numero <= 0.366786) {andarSaida = 11;}
if(numero > 0.366786 && numero <= 0.405008) {andarSaida = 12;}
if(numero > 0.405008 && numero <= 0.429109) {andarSaida = 13;}
if(numero > 0.429109 && numero <= 0.506119) {andarSaida = 17;}
if(numero > 0.506119 && numero <= 0.543777) {andarSaida = 18;}
if(numero > 0.543777 && numero <= 0.588401) {andarSaida = 19;}
if(numero > 0.588401 && numero <= 0.625683) {andarSaida = 20;}
if(numero > 0.625683 && numero <= 0.711542) {andarSaida = 21;}
if(numero > 0.711542 && numero <= 0.763133) {andarSaida = 22;}
if(numero > 0.763133 && numero <= 0.845039) {andarSaida = -1;}
if(numero > 0.845039 && numero <= 0.867445) {andarSaida = -3;}
if(numero > 0.867445 && numero <= 0.942949) {andarSaida = -4;}
if(numero > 0.942949 && numero <= 1) {andarSaida = 0;}
break;

case 16:
if(numero <= 0.039318) {andarSaida = 1;}
if(numero > 0.039318 && numero <= 0.079933) {andarSaida = 2;}
if(numero > 0.079933 && numero <= 0.163761) {andarSaida = 3;}
if(numero > 0.163761 && numero <= 0.179711) {andarSaida = 6;}
if(numero > 0.179711 && numero <= 0.193249) {andarSaida = 8;}
if(numero > 0.193249 && numero <= 0.266691) {andarSaida = 9;}
if(numero > 0.266691 && numero <= 0.314540) {andarSaida = 10;}
if(numero > 0.314540 && numero <= 0.385200) {andarSaida = 11;}
if(numero > 0.385200 && numero <= 0.446402) {andarSaida = 17;}
if(numero > 0.446402 && numero <= 0.551558) {andarSaida = 18;}
if(numero > 0.551558 && numero <= 0.631306) {andarSaida = 19;}
if(numero > 0.631306 && numero <= 0.661350) {andarSaida = 20;}
if(numero > 0.661350 && numero <= 0.751298) {andarSaida = 21;}
if(numero > 0.751298 && numero <= 0.830304) {andarSaida = 22;}
if(numero > 0.830304 && numero <= 0.878524) {andarSaida = -1;}
if(numero > 0.878524 && numero <= 0.918398) {andarSaida = -2;}
if(numero > 0.918398 && numero <= 0.938056) {andarSaida = -3;}
if(numero > 0.938056 && numero <= 1) {andarSaida = 0;}
break;

case 17:
if(numero <= 0.034305) {andarSaida = 1;}
if(numero > 0.034305 && numero <= 0.068420) {andarSaida = 2;}
if(numero > 0.068420 && numero <= 0.145988) {andarSaida = 3;}
if(numero > 0.145988 && numero <= 0.211549) {andarSaida = 4;}
if(numero > 0.211549 && numero <= 0.269487) {andarSaida = 5;}
if(numero > 0.269487 && numero <= 0.300362) {andarSaida = 6;}
if(numero > 0.300362 && numero <= 0.329522) {andarSaida = 7;}
if(numero > 0.329522 && numero <= 0.418334) {andarSaida = 9;}
if(numero > 0.418334 && numero <= 0.442729) {andarSaida = 10;}
if(numero > 0.442729 && numero <= 0.466171) {andarSaida = 11;}
if(numero > 0.466171 && numero <= 0.468077) {andarSaida = 12;}
if(numero > 0.468077 && numero <= 0.502192) {andarSaida = 13;}
if(numero > 0.502192 && numero <= 0.505241) {andarSaida = 15;}
if(numero > 0.505241 && numero <= 0.546026) {andarSaida = 18;}
if(numero > 0.546026 && numero <= 0.611588) {andarSaida = 19;}
if(numero > 0.611588 && numero <= 0.685916) {andarSaida = 20;}
if(numero > 0.685916 && numero <= 0.735277) {andarSaida = 21;}
if(numero > 0.735277 && numero <= 0.760244) {andarSaida = 22;}
if(numero > 0.760244 && numero <= 0.821612) {andarSaida = -1;}
if(numero > 0.821612 && numero <= 0.879741) {andarSaida = -2;}
if(numero > 0.879741 && numero <= 0.903183) {andarSaida = -3;}
if(numero > 0.903183 && numero <= 0.940347) {andarSaida = -4;}
```

```
if(numero > 0.940347 && numero <= 1) {andarSaida = 0;}
break;

case 18:
if(numero <= 0.061019) {andarSaida = 1;}
if(numero > 0.061019 && numero <= 0.104684) {andarSaida = 2;}
if(numero > 0.104684 && numero <= 0.153200) {andarSaida = 3;}
if(numero > 0.153200 && numero <= 0.236238) {andarSaida = 4;}
if(numero > 0.236238 && numero <= 0.267214) {andarSaida = 5;}
if(numero > 0.267214 && numero <= 0.332525) {andarSaida = 6;}
if(numero > 0.332525 && numero <= 0.376190) {andarSaida = 7;}
if(numero > 0.376190 && numero <= 0.386639) {andarSaida = 9;}
if(numero > 0.386639 && numero <= 0.434409) {andarSaida = 10;}
if(numero > 0.434409 && numero <= 0.455122) {andarSaida = 11;}
if(numero > 0.455122 && numero <= 0.532935) {andarSaida = 13;}
if(numero > 0.532935 && numero <= 0.562045) {andarSaida = 15;}
if(numero > 0.562045 && numero <= 0.579959) {andarSaida = 16;}
if(numero > 0.579959 && numero <= 0.604590) {andarSaida = 17;}
if(numero > 0.604590 && numero <= 0.623810) {andarSaida = 19;}
if(numero > 0.623810 && numero <= 0.653480) {andarSaida = 20;}
if(numero > 0.653480 && numero <= 0.723083) {andarSaida = 21;}
if(numero > 0.723083 && numero <= 0.794551) {andarSaida = 22;}
if(numero > 0.794551 && numero <= 0.863407) {andarSaida = -1;}
if(numero > 0.863407 && numero <= 0.888785) {andarSaida = -2;}
if(numero > 0.888785 && numero <= 0.932637) {andarSaida = -3;}
if(numero > 0.932637 && numero <= 1) {andarSaida = 0;}
break;

case 19:
if(numero <= 0.050818) {andarSaida = 1;}
if(numero > 0.050818 && numero <= 0.118212) {andarSaida = 2;}
if(numero > 0.118212 && numero <= 0.181679) {andarSaida = 3;}
if(numero > 0.181679 && numero <= 0.259324) {andarSaida = 4;}
if(numero > 0.259324 && numero <= 0.277863) {andarSaida = 5;}
if(numero > 0.277863 && numero <= 0.336314) {andarSaida = 6;}
if(numero > 0.336314 && numero <= 0.347437) {andarSaida = 7;}
if(numero > 0.347437 && numero <= 0.364013) {andarSaida = 9;}
if(numero > 0.364013 && numero <= 0.422465) {andarSaida = 10;}
if(numero > 0.422465 && numero <= 0.436205) {andarSaida = 11;}
if(numero > 0.436205 && numero <= 0.481352) {andarSaida = 12;}
if(numero > 0.481352 && numero <= 0.518212) {andarSaida = 13;}
if(numero > 0.518212 && numero <= 0.612214) {andarSaida = 15;}
if(numero > 0.612214 && numero <= 0.618975) {andarSaida = 18;}
if(numero > 0.618975 && numero <= 0.629880) {andarSaida = 20;}
if(numero > 0.629880 && numero <= 0.750927) {andarSaida = 21;}
if(numero > 0.750927 && numero <= 0.803272) {andarSaida = 22;}
if(numero > 0.803272 && numero <= 0.867176) {andarSaida = -1;}
if(numero > 0.867176 && numero <= 0.913850) {andarSaida = -2;}
if(numero > 0.913850 && numero <= 0.915812) {andarSaida = -3;}
if(numero > 0.915812 && numero <= 0.937841) {andarSaida = -4;}
if(numero > 0.937841 && numero <= 1) {andarSaida = 0;}
break;

case 20:
if(numero <= 0.050321) {andarSaida = 1;}
if(numero > 0.050321 && numero <= 0.113658) {andarSaida = 3;}
if(numero > 0.113658 && numero <= 0.161648) {andarSaida = 4;}
if(numero > 0.161648 && numero <= 0.220711) {andarSaida = 5;}
if(numero > 0.220711 && numero <= 0.271032) {andarSaida = 6;}
```

```
if(numero > 0.271032 && numero <= 0.297261) {andarSaida = 7;}
if(numero > 0.297261 && numero <= 0.300563) {andarSaida = 8;}
if(numero > 0.300563 && numero <= 0.352050) {andarSaida = 10;}
if(numero > 0.352050 && numero <= 0.384107) {andarSaida = 11;}
if(numero > 0.384107 && numero <= 0.469594) {andarSaida = 12;}
if(numero > 0.469594 && numero <= 0.475617) {andarSaida = 13;}
if(numero > 0.475617 && numero <= 0.498349) {andarSaida = 15;}
if(numero > 0.498349 && numero <= 0.512532) {andarSaida = 16;}
if(numero > 0.512532 && numero <= 0.525549) {andarSaida = 17;}
if(numero > 0.525549 && numero <= 0.532932) {andarSaida = 18;}
if(numero > 0.532932 && numero <= 0.610647) {andarSaida = 19;}
if(numero > 0.610647 && numero <= 0.631824) {andarSaida = 21;}
if(numero > 0.631824 && numero <= 0.730523) {andarSaida = 22;}
if(numero > 0.730523 && numero <= 0.812512) {andarSaida = -1;}
if(numero > 0.812512 && numero <= 0.841850) {andarSaida = -2;}
if(numero > 0.841850 && numero <= 0.894890) {andarSaida = -3;}
if(numero > 0.894890 && numero <= 0.919759) {andarSaida = -4;}
if(numero > 0.919759 && numero <= 1) {andarSaida = 0;}
break;

case 21:
if(numero <= 0.048605) {andarSaida = 1;}
if(numero > 0.048605 && numero <= 0.075101) {andarSaida = 2;}
if(numero > 0.075101 && numero <= 0.111248) {andarSaida = 3;}
if(numero > 0.111248 && numero <= 0.170732) {andarSaida = 6;}
if(numero > 0.170732 && numero <= 0.232848) {andarSaida = 7;}
if(numero > 0.232848 && numero <= 0.258993) {andarSaida = 8;}
if(numero > 0.258993 && numero <= 0.288121) {andarSaida = 9;}
if(numero > 0.288121 && numero <= 0.293209) {andarSaida = 10;}
if(numero > 0.293209 && numero <= 0.337077) {andarSaida = 12;}
if(numero > 0.337077 && numero <= 0.417968) {andarSaida = 13;}
if(numero > 0.417968 && numero <= 0.451307) {andarSaida = 14;}
if(numero > 0.451307 && numero <= 0.524653) {andarSaida = 15;}
if(numero > 0.524653 && numero <= 0.576943) {andarSaida = 16;}
if(numero > 0.576943 && numero <= 0.651167) {andarSaida = 17;}
if(numero > 0.651167 && numero <= 0.695210) {andarSaida = 18;}
if(numero > 0.695210 && numero <= 0.709949) {andarSaida = 19;}
if(numero > 0.709949 && numero <= 0.713634) {andarSaida = 20;}
if(numero > 0.713634 && numero <= 0.755045) {andarSaida = 22;}
if(numero > 0.755045 && numero <= 0.826461) {andarSaida = -1;}
if(numero > 0.826461 && numero <= 0.884015) {andarSaida = -2;}
if(numero > 0.884015 && numero <= 0.898403) {andarSaida = -3;}
if(numero > 0.898403 && numero <= 0.948587) {andarSaida = -4;}
if(numero > 0.948587 && numero <= 1) {andarSaida = 0;}
break;

case 22:
if(numero <= 0.051318) {andarSaida = 1;}
if(numero > 0.051318 && numero <= 0.066685) {andarSaida = 2;}
if(numero > 0.066685 && numero <= 0.125758) {andarSaida = 3;}
if(numero > 0.125758 && numero <= 0.158748) {andarSaida = 4;}
if(numero > 0.158748 && numero <= 0.202030) {andarSaida = 5;}
if(numero > 0.202030 && numero <= 0.251234) {andarSaida = 6;}
if(numero > 0.251234 && numero <= 0.284365) {andarSaida = 7;}
if(numero > 0.284365 && numero <= 0.320598) {andarSaida = 8;}
if(numero > 0.320598 && numero <= 0.372762) {andarSaida = 9;}
if(numero > 0.372762 && numero <= 0.410405) {andarSaida = 10;}
if(numero > 0.410405 && numero <= 0.453828) {andarSaida = 11;}
if(numero > 0.453828 && numero <= 0.499789) {andarSaida = 12;}
```

```
        if(numero > 0.499789 && numero <= 0.519244) {andarSaida = 13;}
        if(numero > 0.519244 && numero <= 0.529254) {andarSaida = 14;}
        if(numero > 0.529254 && numero <= 0.588468) {andarSaida = 15;}
        if(numero > 0.588468 && numero <= 0.617792) {andarSaida = 16;}
        if(numero > 0.617792 && numero <= 0.657409) {andarSaida = 17;}
        if(numero > 0.657409 && numero <= 0.708163) {andarSaida = 18;}
        if(numero > 0.708163 && numero <= 0.733540) {andarSaida = 19;}
        if(numero > 0.733540 && numero <= 0.747498) {andarSaida = 20;}
        if(numero > 0.747498 && numero <= 0.767799) {andarSaida = 21;}
        if(numero > 0.767799 && numero <= 0.819540) {andarSaida = -1;}
        if(numero > 0.819540 && numero <= 0.852531) {andarSaida = -2;}
        if(numero > 0.852531 && numero <= 0.894544) {andarSaida = -3;}
        if(numero > 0.894544 && numero <= 0.950656) {andarSaida = -4;}
        if(numero > 0.950656 && numero <= 1) {andarSaida = 0;}
        break;

}

return andarSaida;
```

ANEXO 2 – Tabela de probabilidades para cada andar de saída

	Andares de Entrada					
Andar de saída	**4S (-4)**	**3S (-3)**	**2S (-2)**	**1S (-1)**	**Térreo (0)**	**1**
4S (-4)	-	-	0,002441	0,038170	0,034818	0,032118
3S (-3)	0,016750	-	0,003051	0,050177	0,029953	0,036111
2S (-2)	0,009678	0,005173	-	0,005688	-	0,057118
1S (-1)	0,080588	0,069911	0,023596	-	0,014748	0,004861
Térreo (0)	0,079844	0,046560	0,054312	0,026163	-	0,024132
1	0,037409	0,075923	0,036412	0,062437	0,028280	-
2	0,077983	0,023070	0,014849	0,044869	0,035426	0,036632
3	0,054718	0,039569	0,061229	0,066355	0,054432	0,026389
4	0,085241	0,039989	0,015663	0,045753	0,065075	0,024653
5	0,032012	0,033138	0,026241	0,050430	0,053216	
6	0,051554	0,033977	0,050651	0,039939	0,047134	0,064063
7	0,045784	0,056348	0,082181	0,053842	0,047134	0,056076
8	-	0,023490	-	0,033367	0,039684	0,009201
9	0,015448	0,053691	0,024003	0,037285	0,025544	-
10	0,025498	0,048658	0,029699	0,037917	0,041356	0,065972
11	0,018425	0,052293	0,039056	0,042594	0,038619	0,057639
12	0,039829	0,049357	0,059805	0,033746	0,031625	0,068576
13	0,035920	0,039989	0,058584	0,042341	0,050631	0,069444
14	-	0,008809	0,018714	0,011122	-	0,020486
15	0,051182	0,044603	0,015053	0,038170	0,037859	0,072049
16	0,005025	0,049357	0,066924	0,027300	0,051999	0,028299
17	0,060860	0,041806	0,072417	0,038928	0,053368	0,007639
18	0,037037	0,042506	0,042514	0,033873	0,031625	0,072743
19	0,010422	0,036353	0,049837	0,037917	0,052151	0,040104
20	0,072213	0,015101	0,023800	0,033620	0,048958	-
21	-	0,016079	0,055126	0,036021	0,027368	0,049826
22	0,056579	0,054251	0,073841	0,031977	0,058993	0,059201

	Andares de Entrada					
Andar de saída	**2**	**3**	**4**	**5**	**6**	**7**
4S (-4)	0,043362	0,041017	0,085066	0,003082	0,058267	0,052295
3S (-3)	0,045697	0,034220	0,055267	0,079373	0,051215	0,091802
2S (-2)	0,014343	0,020274	0,010742	0,034164	0,041937	0,052295
1S (-1)	0,086724	0,035978	0,081428	0,075263	0,083689	0,085179
Térreo (0)	0,016678	0,043009	0,030319	0,066787	0,062720	0,079927
1	0,016678	0,039845	0,062370	0,008220	0,052329	0,057091
2	-	0,026485	-	-	0,002969	0,015300
3	0,034857	-	0,052148	-	0,102245	0,068280
4	0,011341	0,016758	-	0,006422	-	-
5	0,027685	0,033869	-	-	-	-
6	0,045364	0,045353	0,031185	-	-	-
7	0,045364	0,058244	-	0,004110	-	-
8	0,002168	0,056838	0,021483	-	0,016515	-
9	0,081888	0,054260	0,049030	0,029797	0,050659	0,041334
10	0,071881	0,041134	0,041753	-	0,020226	0,101393
11	0,056204	0,055315	0,034650	0,082969	0,016144	-
12	0,068879	0,051330	0,068087	0,067557	0,087029	-
13	0,055537	0,030353	0,055440	0,111482	0,021154	0,103448
14	0,012008	-	-	-	0,014474	-
15	0,062542	0,051213	0,025295	0,018238	0,035814	0,083124
16	0,003336	0,028243	0,055267	0,093244	0,055112	-
17	0,027185	0,038439	0,029799	0,033136	0,015587	0,051382
18	0,046698	0,045588	0,052495	0,100950	0,048803	0,006623
19	0,025017	0,036916	0,012474	0,033393	0,021340	-
20	0,015677	0,031876	0,068434	0,051374	0,038968	0,004111
21	0,051201	0,042892	0,026334	0,013614	0,056226	0,039507
22	0,031688	0,040548	0,050936	0,086823	0,046576	0,066910

	Andares de Entrada					
Andar de saída	**8**	**9**	**10**	**11**	**12**	**13**
4S (-4)	0,053938	0,089185	-	-	0,051212	0,016786
3S (-3)	0,028253	0,075634	0,075283	0,050344	0,104671	0,082267
2S (-2)	0,077911	0,080315	-	0,054671	0,007744	0,045704
1S (-1)	0,081621	0,095590	0,085499	0,070403	0,092431	0,068140
Térreo (0)	0,130137	0,059867	0,051743	0,038545	0,072446	0,044208
1	0,101884	0,056664	0,069509	0,045821	0,089683	0,054845
2	-	0,049273	0,003553	0,004130	0,043967	0,019445
3	0,100742	0,089677	0,057739	0,061947	0,114914	0,071464
4	-	-	0,026205	0,022419	0,069698	0,051022
5	-	-	-	0,070796	0,078441	0,01188
6	-	0,011579	0,107706	0,023795	0,041469	0,080605
7	-	0,051491	0,012214	0,020846	0,019485	0,056507
8	-	-	-	-	-	0,071963
9	-	-	0,034866	0,020059	-	0,067974
10	-	0,029071	-	-	-	0,010969
11	-	-	0,005996	-	0,018236	-
12	0,151826	0,010594	-	0,016716	-	-
13	0,123002	-	0,037975	0,007473	-	-
14	-	-	-	-	-	0,004487
15	-	0,030303	0,037308	0,049164	0,044467	0,029583
16	-	0,038679	-	0,095182	0,018986	-
17	0,047374	0,022173	0,057961	0,059587	0,008244	0,042048
18	-	0,068736	0,086387	0,016716	-	0,035898
19	-	0,058389	0,062847	0,059390	-	0,026924
20	-	-	0,084388	0,073156	-	-
21	0,019977	0,023898	0,031535	0,075910	0,085936	0,028752
22	0,083333	0,058881	0,071286	0,062930	0,037972	0,039222

	Andares de Entrada					
Andar de saída	**14**	**15**	**16**	**17**	**18**	**19**
4S (-4)	-	0,075504	-	0,037164	-	0,022028
3S (-3)	0,275410	0,022406	0,019659	0,023442	0,043851	0,001963
2S (-2)	-	-	0,039874	0,058128	0,025378	0,046674
1S (-1)	0,072131	0,081905	0,048220	0,061368	0,068856	0,063904
Térreo (0)	0,232787	0,057051	0,061944	0,059653	0,067363	0,062159
1	-	0,015440	0,039318	0,034305	0,061019	0,050818
2	-	0,042930	0,040616	0,034115	0,043665	0,067394
3	-	0,078516	0,083828	0,077568	0,048517	0,063468
4	-	0,041047	-	0,065561	0,083038	0,077644
5	-	-	-	0,057938	0,030976	0,018539
6	0,121311	0,069102	0,015950	0,030875	0,065311	0,058451
7	-	0,005460	-	0,029160	0,043665	0,011123
8	-	-	0,013539	-	-	-
9	-	-	0,073442	0,088813	0,010450	0,016576
10	-	0,095462	0,047849	0,024395	0,047770	0,058451
11	-	0,018829	0,070660	0,023442	0,020713	0,013740
12	-	0,038223	-	0,001906	-	0,045147
13	-	0,024101	-	0,034115	0,077813	0,036859
14	-	-	-	-	-	-
15	-	-	-	0,003049	0,029110	0,094002
16	-	-	-	-	0,017914	-
17	0,022951	0,077010	0,061202	-	0,024631	-
18	0,275410	0,037658	0,105156	0,040785	-	0,006761
19	-	0,044624	0,079748	0,065561	0,019220	-
20	-	0,037281	0,030045	0,074328	0,029670	0,010905
21	-	0,085860	0,089948	0,049362	0,069603	0,121047
22	-	0,051591	0,079006	0,024967	0,071469	0,052345

	Andares de Entrada		
Andar de saída	**20**	**21**	**22**
4S (-4)	0,024869	0,050184	0,056112
3S (-3)	0,053041	0,014388	0,042013
2S (-2)	0,029337	0,057554	0,032990
1S (-1)	0,081990	0,071416	0,051741
Térreo (0)	0,080241	0,051413	0,049344
1	0,050321	0,048605	0,051318
2	-	0,026496	0,015367
3	0,063338	0,036147	0,059072
4	0,047989	-	0,032990
5	0,059064	-	0,043282
6	0,050321	0,059484	0,049203
7	0,026229	0,062116	0,033131
8	0,003303	0,026145	0,036233
9	-	0,029128	0,052164
10	0,051486	0,005089	0,037643
11	0,032058	-	0,043423
12	0,085487	0,043867	0,045961
13	0,006023	0,080891	0,019456
14	-	0,033339	0,010010
15	0,022732	0,073346	0,059213
16	0,014183	0,053390	0,029325
17	0,013017	0,074224	0,039617
18	0,007383	0,044043	0,050754
19	0,077715	0,014739	0,025377
20	-	0,003685	0,013957
21	0,021177	-	0,020302
22	0,098698	0,041411	-