UFMG – DCC – Algoritmos e Estruturas de Dados II - 1º Semestre de 2017
Trabalho Prático 1 – Fila, Pilhas e Complexidade

Professor: Adriano Veloso
Aluno: Paulo Fontes Mol Júnior - Matrícula: 2016115186 – Turma: TW

Simulação de fluxo de atendimento na cantina do ICEX – AEDs II

1 – Introdução:

A cantina do ICEX recebe, diariamente, no horário de almoço um grande número de clientes que devem seguir um fluxo pré-determinado para seja concluído o seu atendimento. Esse fluxo consiste em: ao chegar na cantina, o cliente deve entrar na fila do caixa, aguardar a sua vez de comprar uma ficha no caixa, seguir para uma segunda fila, aguardar a sua vez de pegar uma bandeja com os talheres e então, com a bandeja e os talheres em mãos, seguir para a fila da mesa de alimentos, onde se servirá de 4 tipos de alimentos diferentes. Após cumpridos estes procedimentos, considera-se que o atendimento ao cliente fora concluído.

O problema proposto consiste em simular, computacionalmente, este cenário por um período pré-fixado de 4 horas, a partir do início do horário de almoço, determinar o número total de clientes atendidos nesse período, assim como o tempo médio total de atendimento.

A solução proposta neste trabalho consiste na implementação de três TADs: um responsável pela geração e processamento de fila dinâmica, um responsável pela geração e processamento de pilha estática e um responsável pela geração e processamento da estrutura de dados que representa o cliente. O resultado da simulação total é calculado a partir dos dados contidos em cada cliente que completa o fluxo total.

2 – Desenvolvimento

2.1 - Parâmetros

A solução apresentada neste trabalho se baseia no caso base do enunciado. São implementadas três filas dinâmicas, uma pilha estática e uma estrutura que representa os clientes, ou seja, os dados contidos nas células (ou posições) que serão enfileiradas e desenfileiradas. A maioria das operações é realizada com base na passagem do tempo, que é representado por um laço de repetição que simula a ocorrência de eventos (através de funções e procedimentos) do tempo 0 ao tempo estipulado de 4 horas (240 ciclos). Cada ciclo representa 1 minuto. A cada minuto, 2 novos clientes são enfileirados na primeira fila. A cada 12 minutos, 10 novas posições são somadas ao topo da pilha (até o limite de 30).

Inicialmente foram criadas três filas dinâmicas vazias e uma pilha estática de 30 posições. São denotadas por: Fila1, que representa a fila inicial, ou fila do caixa; Fila2, ou fila das bandejas; Fila3, ou fila dos alimentos e Pilha1, ou pilha de bandejas (cada posição da pilha representa uma bandeja).

2.2 – Estruturas

2.2.1 – Cliente

A estrutura Cliente, ou *TCliente*, contém dos dados que serão inseridos nas células (ou posições) e considerados como parâmetros para o cálculo do resultado. Os dados contidos na estrutura Cliente são: *ID*, que identifica a ordem de chegada do cliente, *Chegada*, que identifica o minuto em que o cliente chega, *Saida*, que identifica o minuto em que o cliente conclui o atendimento e *TempoTotal*, que identifica o número de minutos que o cliente demorou para concluir o atendimento.

2.2.2 – Fila

As estruturas relacionadas com as Filas Dinâmicas são:

Posicao (TPosicao e **Apontador*), relacionada com os os dados contidos em cada posição da fila (células) que contém um *TCliente* e um *Apontador* para a próxima Posição.

Fila (TFila), que determina a fila em si e contém um *Apontador (Posicao) Frente* e um *Apontador (Posicao) Tras*, que apontam, respectivamente, para a primeira e para a última posição da fila.

2.2.3 – Pilha

As estruturas relacionadas com a Pilha Estática são:

Bandejao (TBandeja), que determina um inteiro (*TUtilizadas*) a ser inserido em cada posição da pilha, é utilizada para calcular a quantidade de bandejas utilizadas.

Pilha (TPilha) que determina a pilha em si e contém um *TBandeja*, que limita o número máximo de bandejas na pilha e um *Ponteiro* (*int*) que indica a posição do topo da pilha.

2.3 – Descrição do ciclo

No minuto 0 (*Cronometro* 0), é gerado (inicializado) um cliente (de *ID* 1 e tempo de chegada 0) e este é enfileirado na *Fila1*, que, até então, estava vazia. O cliente de *ID* 1 ocupa, então, a primeira posição da fila do caixa. Na sequência, é gerado um segundo cliente (de *ID* 2 e tempo de chegada 0), que também é enfileirado na *Fila1*. O cliente de *ID* 2 ocupa a segunda posição da *Fila1*. Ainda no tempo 0, como a posição do caixa está vazia, o cliente *ID* 1, primeiro da fila, é copiado para a variável *Caixa* e sua posição na *Fila1* é apagada, de forma que o cliente *ID* 2 passa a ser o primeiro da fila. Simbolicamente, podemos dizer que o cliente *ID* 1 saiu da *Fila1* e se encontra agora no caixa.

Os eventos ocorridos no tempo 0 determinam as condições iniciais do cenário para instaurar a simulação do fluxo de atendimento. Esses eventos denotam o primeiro momento da trajetória de um cliente nesse processo e serão repetidos a cada ciclo, de forma que, a cada minuto (ou a cada ciclo), 2 novos clientes serão enfileirados nas últimas posições da fila do caixa e o cliente da primeira posição é desenfileirado e copiado para a variável *Caixa*. Para cada cliente criado é atribuído

um valor de tempo de chegada (igual ao valor de *Cronômetro*) e uma *ID* (igual à do cliente anterior mais 1).

No minuto 1 (*Cronometro* 1), o valor copiado para *Caixa* no ciclo anterior (cliente de *ID* 1) é enfileirado na *Fila2*, que, até então, estava vazia. O cliente que está na primeira posição da *Fila1* (cliente de *ID* 2), tem seu valor copiado para *Caixa* (Caixa deixa de ser cliente de *ID* 1, que vai para a *Fila2*, e passa a ser o cliente de *ID* 2, que sai da *Fila1*). Como a posição da pilha de bandejas não está ocupada e a pilha de bandejas não está vazia, o cliente da primeira posição da *Fila2* é copiado para a variável *Bandejas*, de forma similar ao que ocorreu na passagem do cliente da *Fila1* para *Caixa* e o topo da pilha é subtraída de 1 posição. Simbolicamente, pode-se dizer que o cliente da primeira posição da *Fila2* saiu da fila e pegou uma bandeja da pilha de bandejas.

Como cada cliente que ‘chega no caixa’ só ‘sai do caixa’ no ciclo seguinte, então temos que o caixa adiciona um minuto ao tempo de atendimento de cada cliente que passa por ele.

No minuto 2 (*Cronometro* 2), o valor copiado para *Bandejas* no ciclo anterior (cliente de *ID* 1) é enfileirado na *Fila3*, que, até então, estava vazia. O cliente de *ID* 2, primeiro da *Fila2*, é desenfileirado e copiado para *Bandejas*, subtrai 1 posição do topo da pilha e o cliente de *ID* 3, na primeira posição da *Fila1*, é desenfileirado e copiado para *Caixa*. Como a posição do primeiro alimento (*Arroz*) não está ocupada, o cliente que ocupa a primeira posição da *Fila3* é desenfileirado e seu valor copiado para *Arroz*. Simbolicamente, podemos dizer que o cliente, já com a bandeja em mãos, foi para a cuba de arroz se servir desse alimento.

Como cada cliente que ‘pega uma bandeja’ só segue para a *Fila3* no ciclo seguinte, temos que a pilha de bandejas adiciona um minuto ao tempo de atendimento de cada cliente que passa por ela (só passará caso a pilha não esteja vazia).

No minuto 3 (*Cronometro* 3), o valor copiado para *Arroz* no ciclo anterior (cliente de *ID* 1), é copiado para o segundo alimento (*Feijao*). O cliente que está na primeira posição da *Fila3* (cliente de *ID* 2) é desenfileirado e seu valor copiado para *Arroz* (caso a *Fila3* estivesse vazia, a posição *Arroz* permaneceria vazia). Nesse ponto, o cliente de *ID* 3 está na pilha de bandejas (cujo topo terá perdido, até então, 3 posições), o cliente de *ID* 4 estará no *Caixa* e o cliente de *ID* 5 estará na primeira posição da *Fila1*.

Como cada cliente que se serve de um alimento só passa para o próximo alimento no ciclo seguinte, temos que cada alimento adiciona 1 minuto no tempo de atendimento de cada cliente que passa por ele.

No minuto 4 (*Cronometro* 4), o valor copiado para *Feijao* no ciclo anterior (cliente de *ID* 1), é copiado para o terceiro alimento (*Guarnicao*) e o valor de *Arroz* (cliente de *ID* 2) é copiado para *Feijao*. O cliente que está na primeira posição da *Fila3* (cliente de *ID* 3) é desenfileirado e seu valor copiado para *Arroz*. Nesse ponto, o cliente de *ID* 4 está na pilha de bandejas (cujo topo terá perdido, até então, 4

posições), o cliente de *ID* 5 estará no *Caixa* e o cliente de *ID* 6 estará na primeira posição da *Fila1*.

No minuto 5 (*Cronometro* 5), o valor copiado para *Guarnicao* no ciclo anterior (cliente de *ID* 1), é copiado para o quarto alimento (*Salada*), o valor de *Feijao* (cliente de *ID* 2) é copiado para *Guarnicao* e o valor de *Arroz* (cliente de *ID* 3) é copiado para *Feijao*. O cliente que está na primeira posição da *Fila3* (cliente de *ID* 4) é desenfileirado e seu valor copiado para *Arroz*. Nesse ponto, o cliente de *ID* 5 está na pilha de bandejas (cujo topo terá perdido, até então, 5 posições), o cliente de *ID* 6 estará no *Caixa* e o cliente de *ID* 7 estará na primeira posição da *Fila1*.

No minuto 6 (*Cronometro* 6), o valor copiado para *Salada* no ciclo anterior (cliente de *ID* 1), é considerado atendido. Seu tempo de saída é definido conforme valor de *Cronometro* e seu tempo total calculado pela subtração do tempo de saída pelo tempo de chegada. O valor copiado para *Guarnicao* no ciclo anterior (cliente de *ID* 2) é copiado para *Salada*, o valor de *Feijao* (cliente de *ID* 3) é copiado para *Guarnicao* e o valor de *Arroz* (cliente de *ID* 4) é copiado para *Feijao*. O cliente que está na primeira posição da *Fila3* (cliente de *ID* 5) é desenfileirado e seu valor copiado para *Arroz*. Nesse ponto, o cliente de *ID* 6 está na pilha de bandejas (cujo topo terá perdido, até então, 6 posições), o cliente de *ID* 7 estará no *Caixa* e o cliente de *ID 8* estará na primeira posição da *Fila1*.

O programa seguirá mantendo este ciclo até que a pilha de bandejas se reduza a 0. Quando isto ocorre, há, temporariamente, um acúmulo de clientes na *Fila2*, uma inibição do crescimento da *Fila3* e um aumento no tempo médio de atendimento.

3 – Implementação

3.1 – Criação de estruturas e inicialização de posições vazias

*void cria_fila(TFila *Fila):* procedimento para criação de uma fila vazia.

*void cria_pilha_cheia(TPilha *Pilha):* procedimento para criação de uma pilha estática cheia

*void bandejas_utilizadas(TBandeja *Bandejas):* inicializa o valor da quantidade de bandejas utilizadas como zero

void define_atraso(TCliente Aguarde): inicializa um *TCliente* com valores -1, criando um TCliente excepcional (inválido) que será utilizado para indicar locais vagos e barrar o fluxo de clientes em determinado ponto e em determinadas situações.

*void atrasa_fila (TCliente Aguarde, TCliente *Bandejas):* inicializa a variável *TCliente* que indica a posição da pilha de bandejas com valores de *TCliente* inválido para que só haja fluxo de clientes a partir dela quando contiver um cliente válido

*void self_service(TCliente Aguarde, TCliente *Arroz, TCliente *Feijao, TCliente *Guarnicao, TCliente *Salada, TCliente *Saida):* inicializa todas as variáveis *TCliente* relacionadas à mesa de alimentos com valores de *TCliente* inválido para que só haja fluxo de clientes quando o alimento anterior contiver um cliente válido.

3.2 – Procedimentos e funções de simulação

*void chega_cliente (TCliente *Cliente, TTempo Chegada, TId ID):* inicializa um cliente válido, com tempo de chegada igual ao valor de Cronometro no momento da chegada. Saida e TempoTotal são inicializados com valor 0. O valor de ID é igual a uma constante mais 2 vezes o tempo de chegada, gerando IDs sequenciais. Essa constante é 1, quando esta função é implementada antes do primeiro enfileiramento e 2 quando implementada depois do primeiro enfileiramento do ciclo.

*void aumenta_fila(TFila *Fila, TCliente Cliente):* adiciona, ao final da fila, um cliente válido proveniente de uma etapa anterior. No caso da fila do caixa, enfileira individualmente os clientes criados; na fila das bandejas, enfileira o cliente que sai do caixa, caso exista; no caso da fila de alimentos, enfileira o cliente que pegou a bandeja, caso exista.

*TCliente atendimento(TFila *Fila, TCliente *Cliente, TCliente Aguarde)*: desenfileira uma posição da fila pela frente e retorna o *TCliente* contido nesta posição, caso haja. Se a fila estiver vazia ou o *TCliente* anterior seja inválido, retorna um *TCliente* inválido para barrar o fluxo da fila temporariamente. É utilizada para atribuir um cliente ao *Caixa* e para atribuir um cliente ao primeiro alimento (*Arroz*).

*void serve_prato(TCliente *Cliente, TCliente *Alimento, TCliente Aguarde):* promove o fluxo de clientes válidos de um alimento para outro. Caso não haja clientes válidos na posição anterior adjacente, mantém a posição vazia e barra o fluxo de clientes naquela posição.

*void finaliza_atendimento(TCliente *Alimento, TCliente *Saida, int *TotalTempo, int *TotalClientes, int Cronometro)*: determina a conclusão do atendimento do cliente, caso o ciclo anterior tenha resultado em um *TCliente* válido na posição do último alimento (*Salada*). Determina o valor *Saida* do cliente, conforme valor do cronômetro, calcula seu tempo total de atendimento subtraindo de *Saida* o valor de *Chegada* e soma esse valor, por referência, ao tempo de atendimento global. Atribui, também por referência, o valor ID de cada usuário atendido, a uma variável do programa principal, de forma que esta variável retenha apenas o valor de ID do último atendido. Como esses valores são sequenciais, o ID do último atendido corresponde ao número total de usuários atendidos.

void imprime_resultado(int TotalClientes, int TotalTempo, TBandeja Utilizadas, int Cronometro): imprime o número total de clientes atendidos, o tempo médio de atendimento, o número de bandejas utilizadas, o período considerado e os parâmetros utilizados.

*void destroi_fila(TFila *Fila):* desaloca toda memória alocada pelas filas.

4 – Análise de complexidade

Foram selecionados algoritmos ótimos para a implementação das estruturas de dados. Filas dinâmicas foram utilizadas para permitir maior flexibilidade à expansão das filas, enquanto a pilha estática se mostrou adequada devido ao número

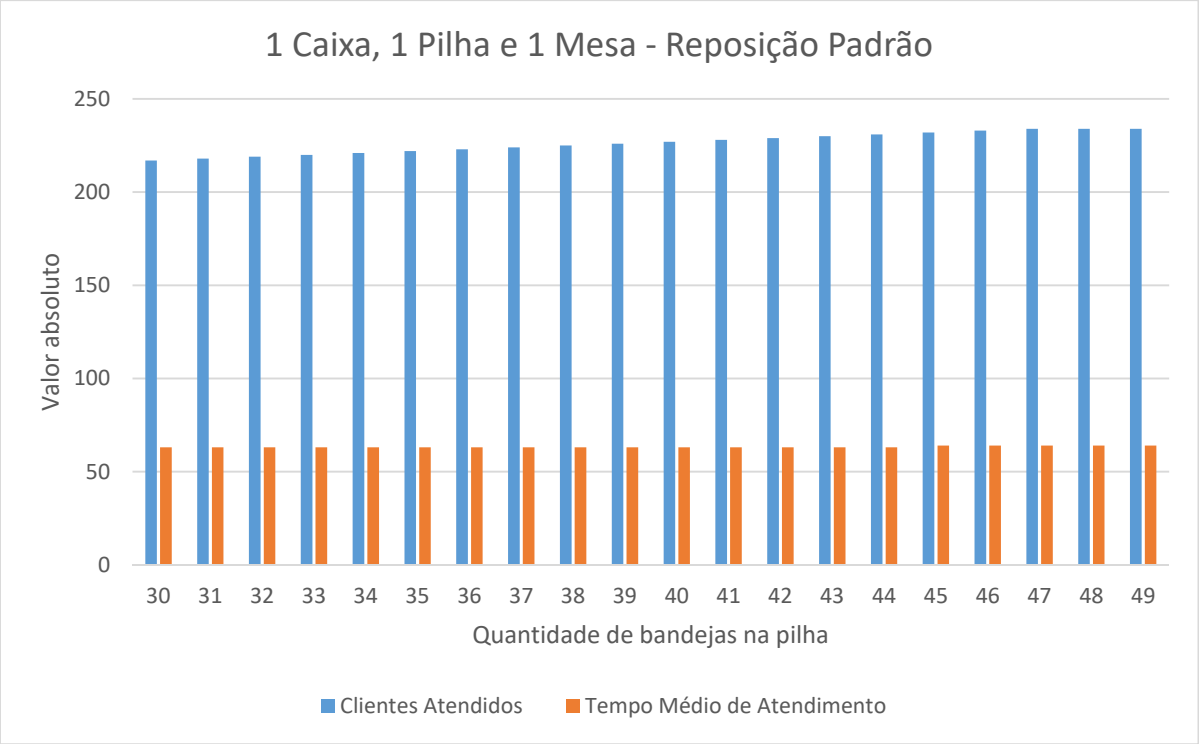
previsível e baixo de elementos empilhados. Como o programa não depende de entrada e os passos dos algoritmos são executados um número fixo de vezes, podemos afirmar que a complexidade dos algoritmos é O(1).

5 – Resultados

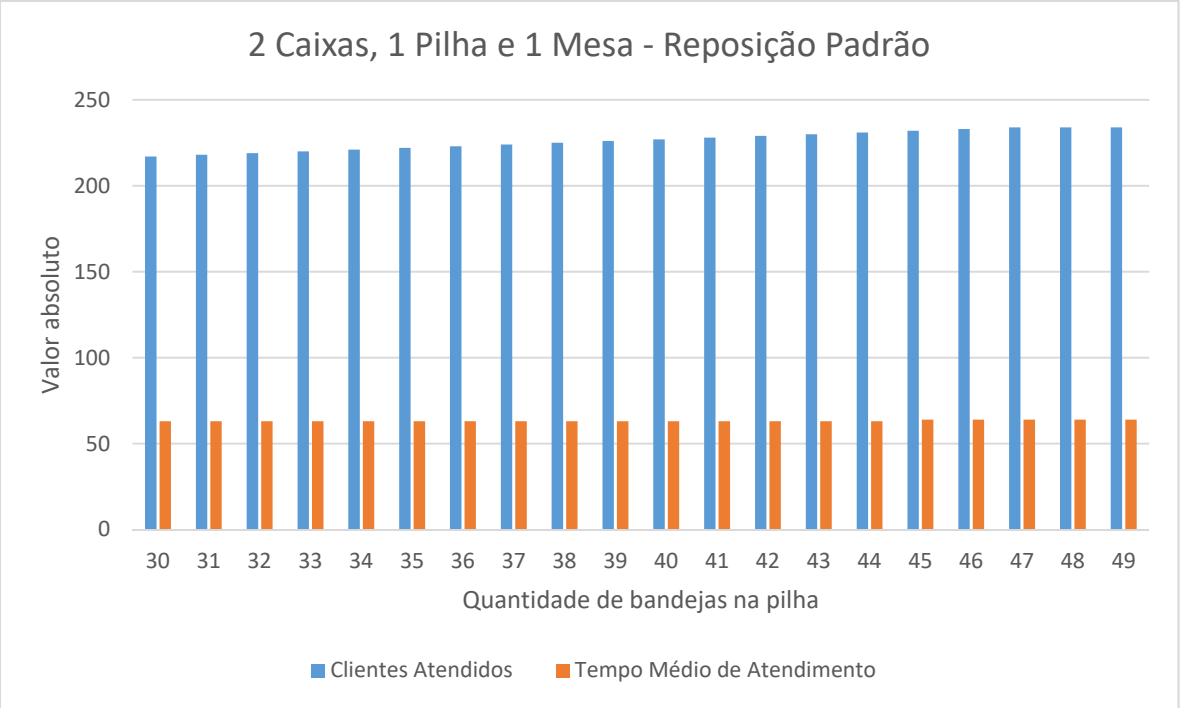
Os parâmetros utilizados no caso base deste trabalho foram: 1 caixa, 1 pilha de bandejas com 30 bandejas, reposição de 10 bandejas a cada 12 minutos e 1 mesa de alimentos com 4 alimentos. Para esta configuração o resultado obtido ao final do período de 4 horas foi de 217 clientes atendidos com um tempo médio de atendimento de 63 minutos

Foram realizadas alterações na configuração e em alguns parâmetros do caso base para averiguar alterações no resultado. As alterações testadas e os novos resultados são ilustrados pelos gráficos abaixo.

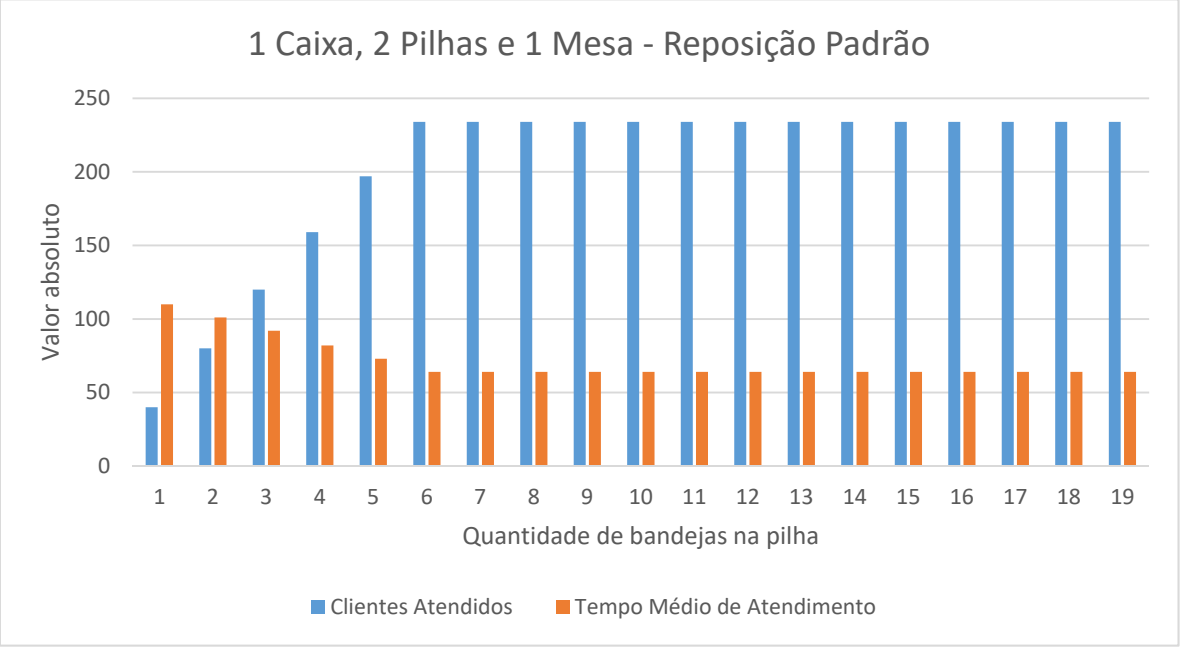
5.1 – Alterando-se a quantidade de bandejas na pilha



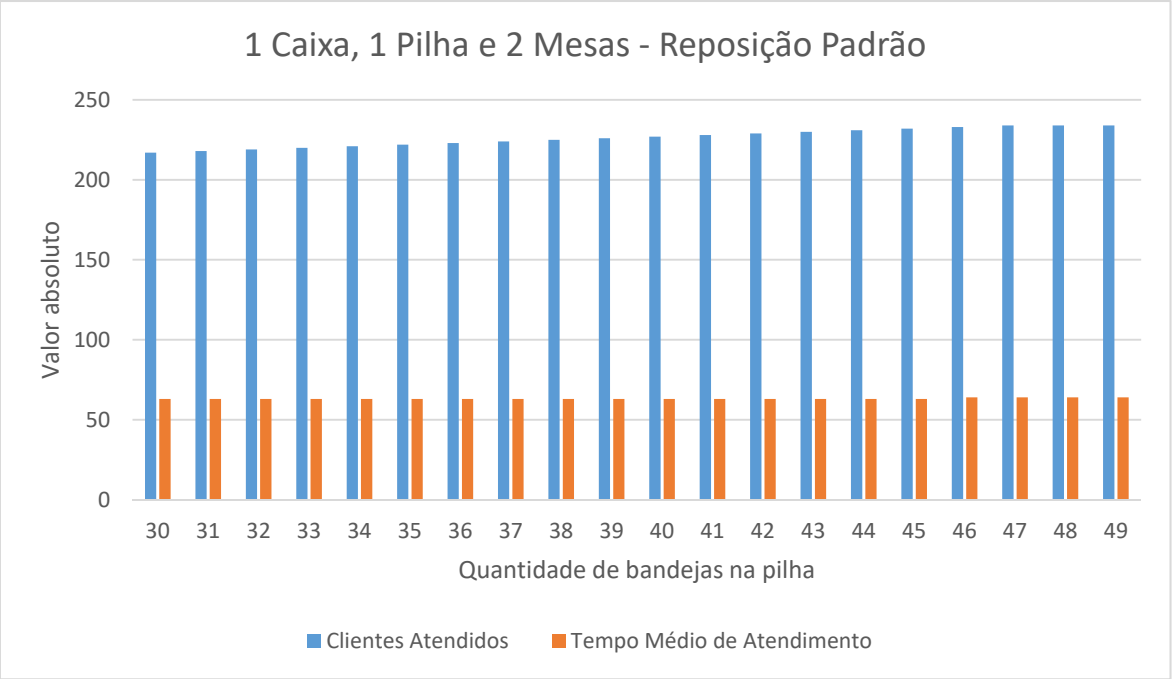
Melhor caso: 47 bandejas (234 clientes atendidos, tempo médio de 64 minutos)



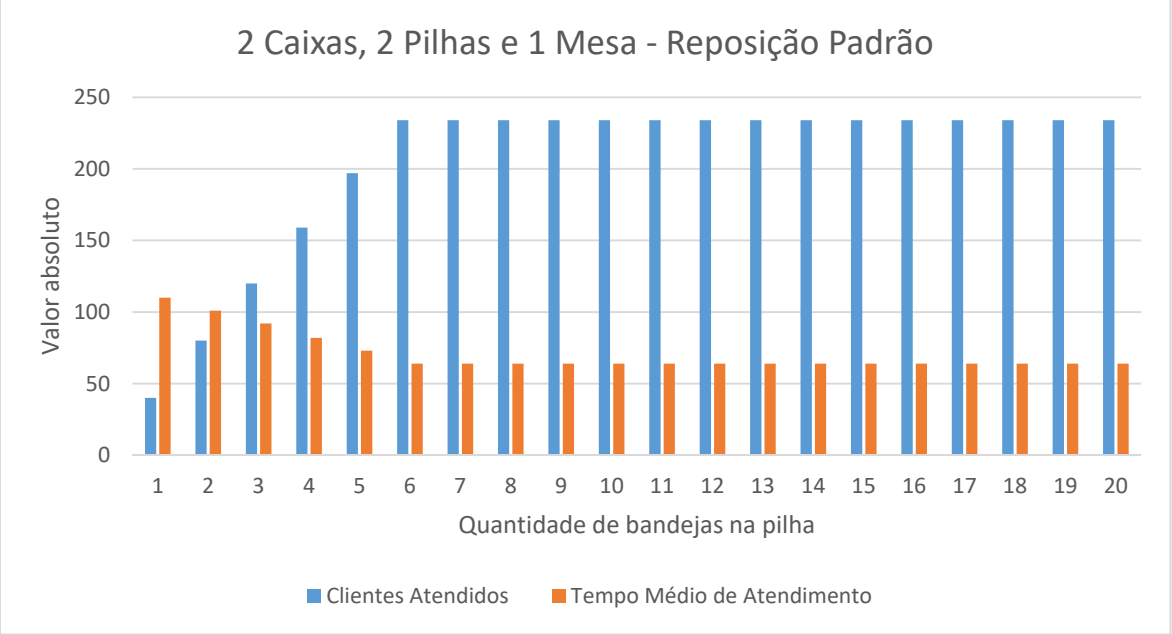
Melhor caso: 47 bandejas (234 clientes atendidos, tempo médio de 64 minutos)



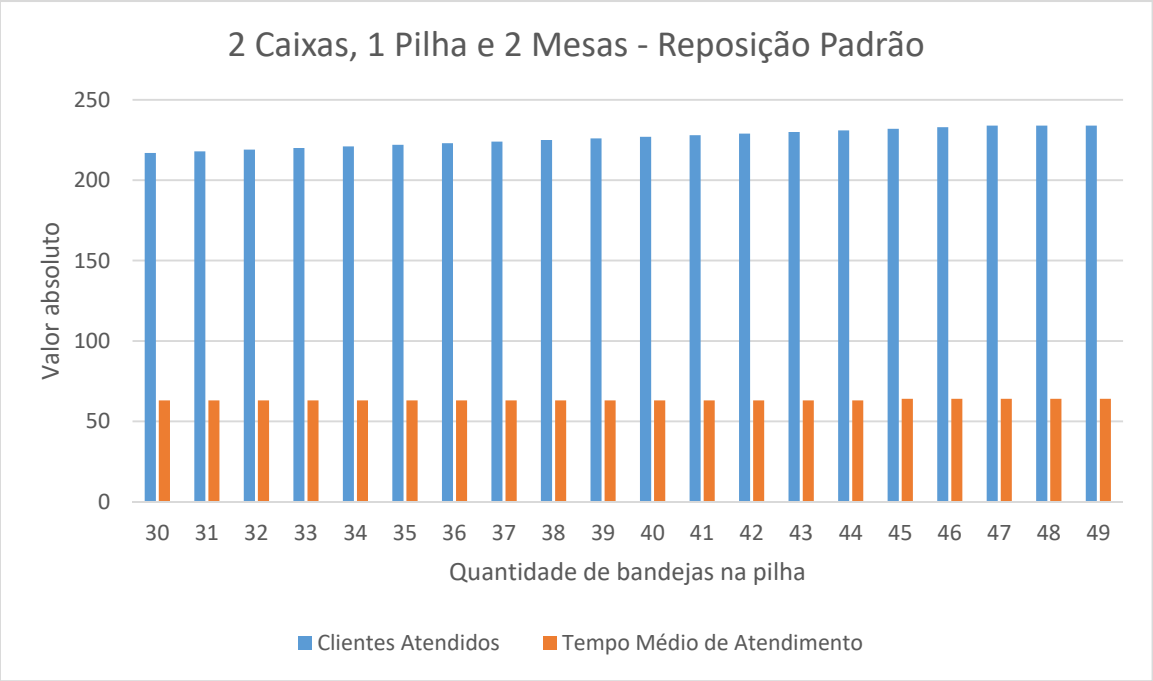
Melhor caso: 6 bandejas (234 clientes atendidos, tempo médio de 64 minutos)



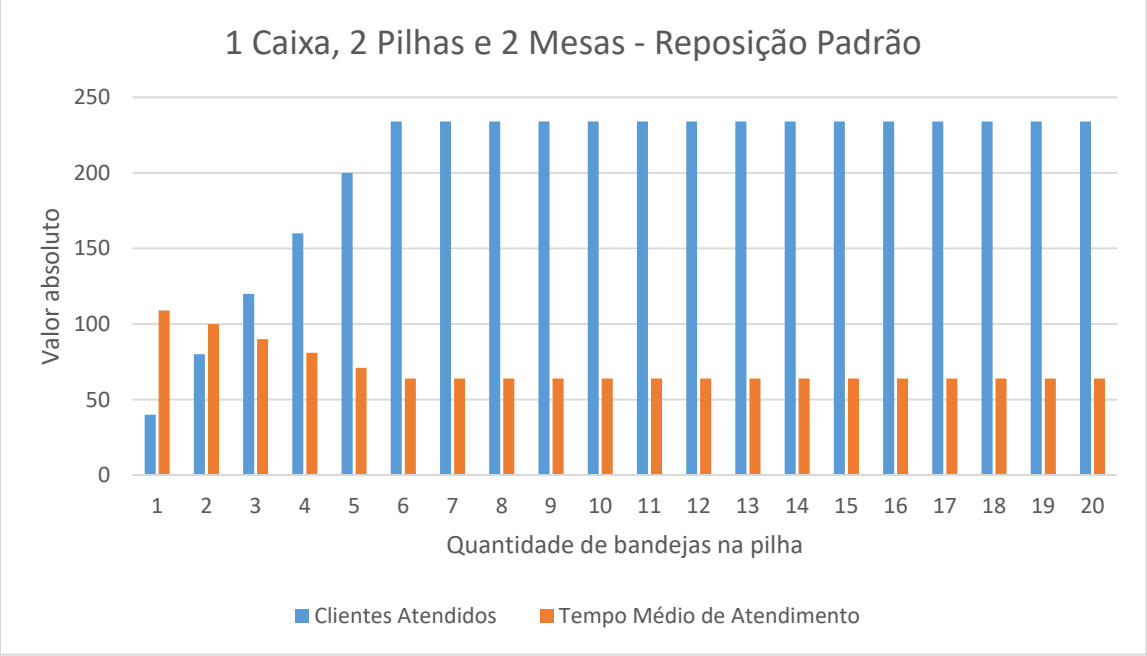
Melhor caso: 47 bandejas (234 clientes atendidos, tempo médio de 64 minutos)



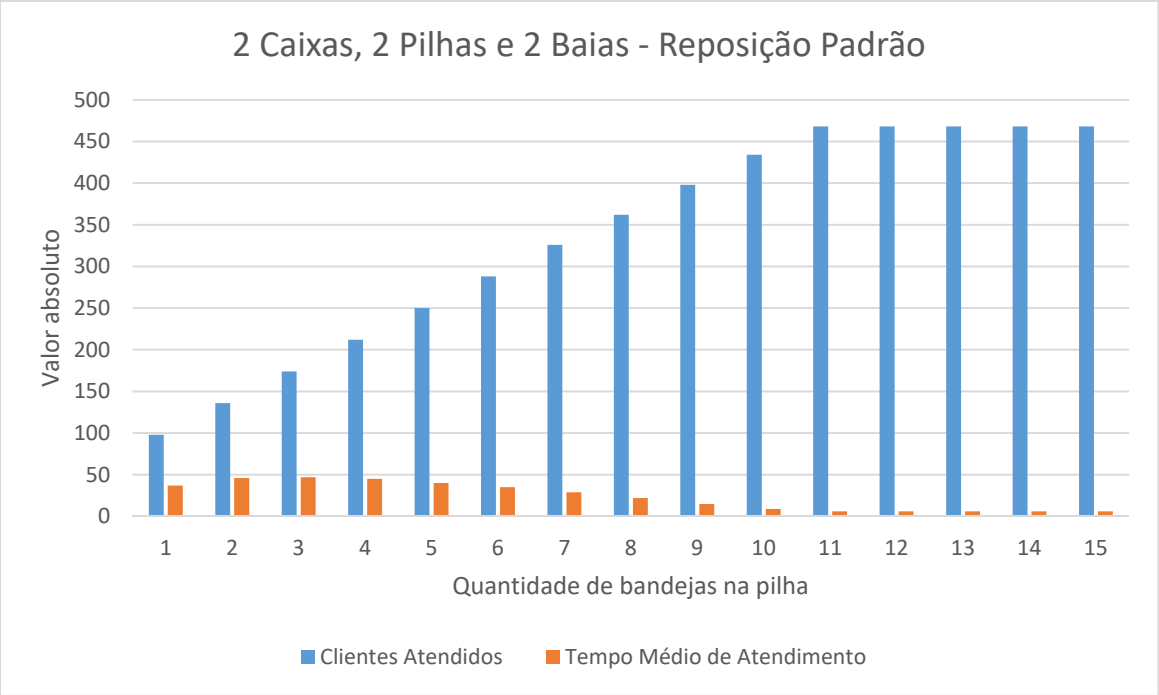
Melhor caso: 6 bandejas (234 clientes atendidos, tempo médio de 64 minutos)



Melhor caso: 47 bandejas (234 clientes atendidos, tempo médio de 64 minutos)

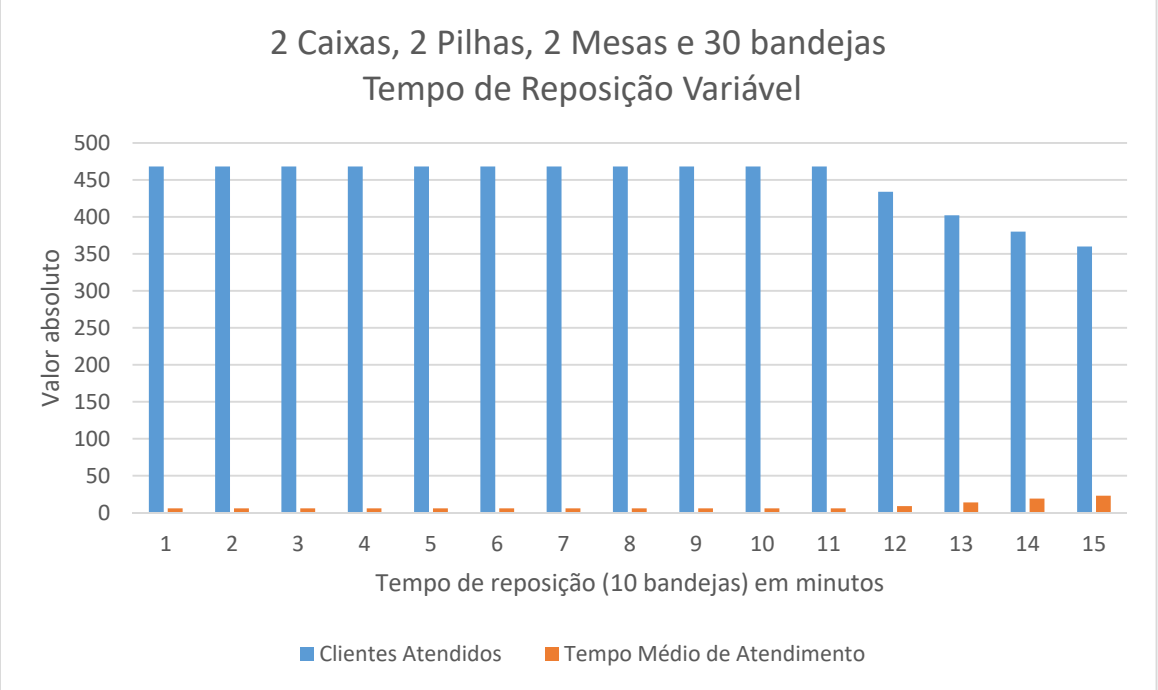


Melhor caso: 6 bandejas (234 clientes atendidos, tempo médio de 64 minutos)

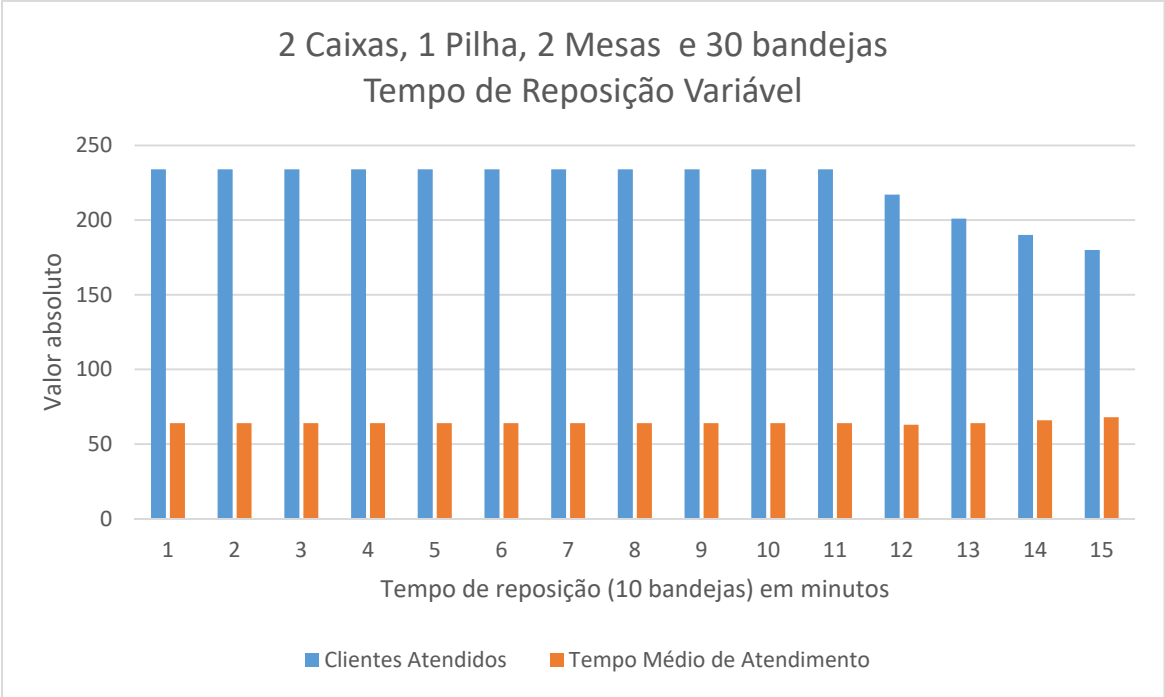


Melhor caso: 11 bandejas (468 clientes atendidos, tempo médio de 6 minutos)

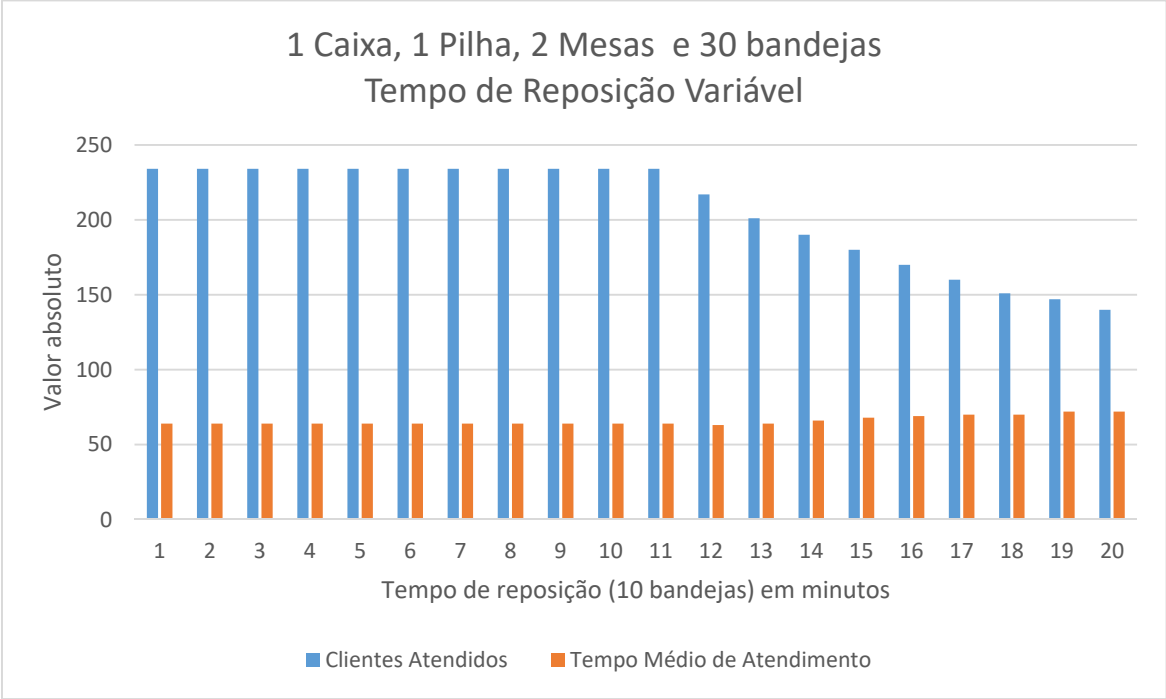
5.2 – Variando-se o tempo de reposição



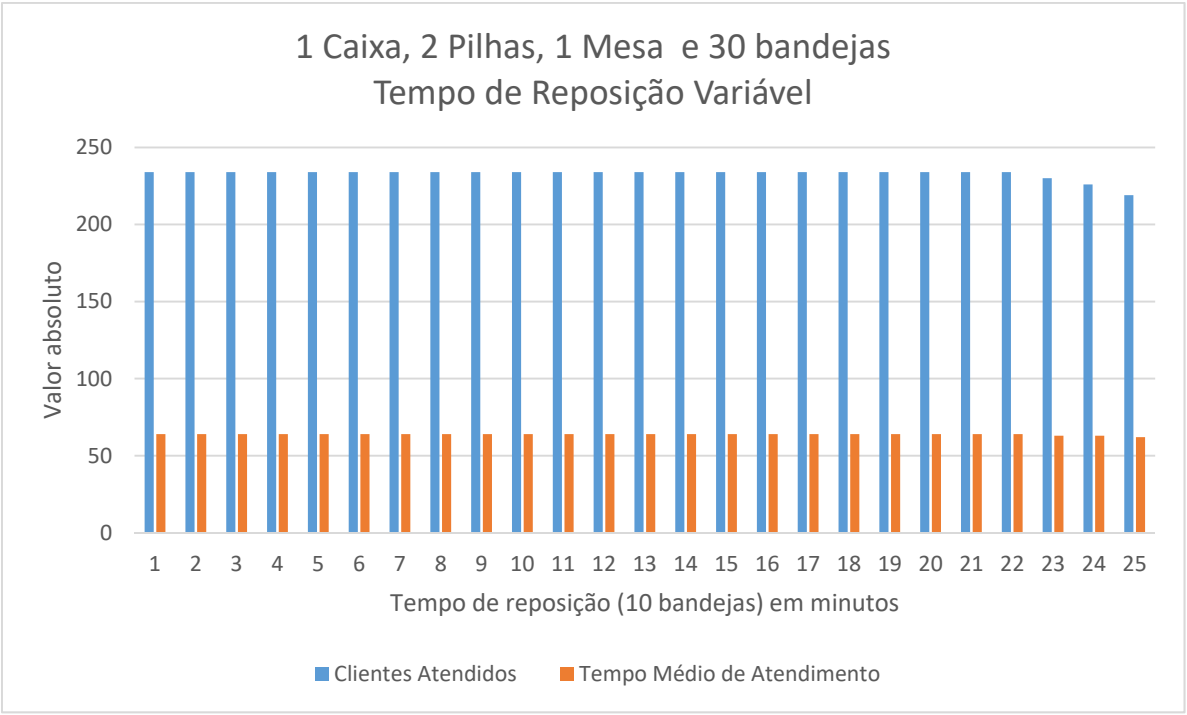
Melhor caso: 11 minutos (468 clientes atendidos, tempo médio de 6 minutos)



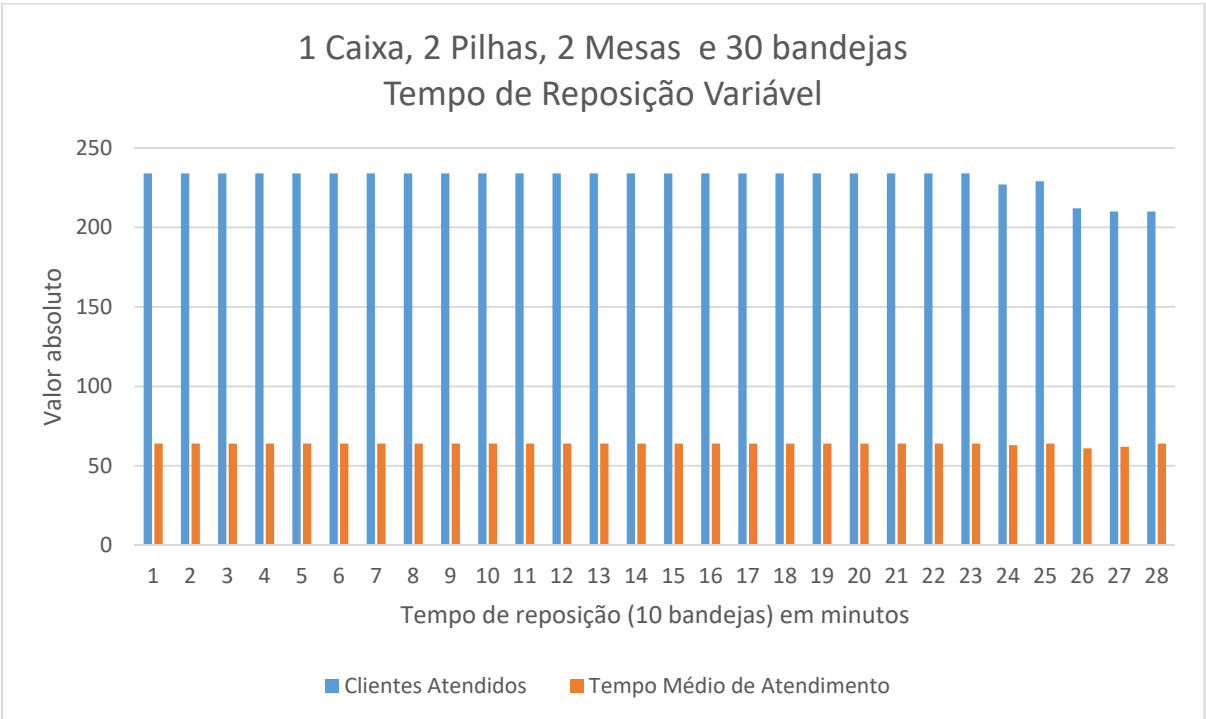
Melhor caso: 11 minutos (234 clientes atendidos, tempo médio de 64 minutos)



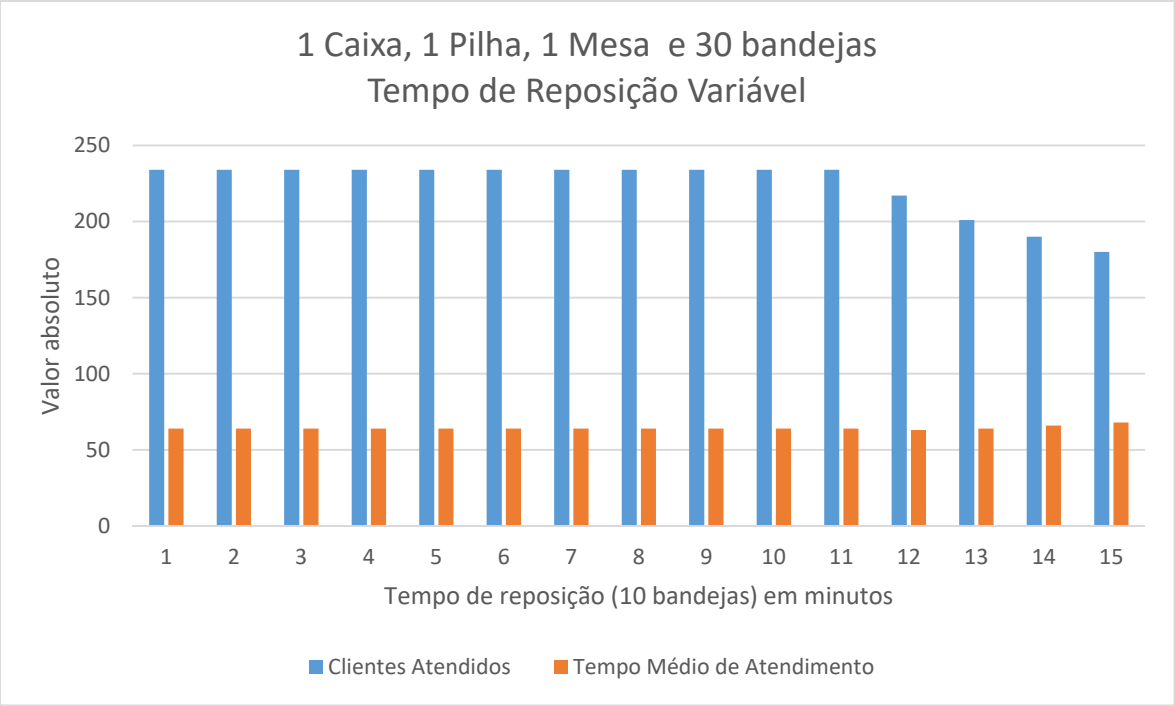
Melhor caso: 11 minutos (234 clientes atendidos, tempo médio de 64 minutos)



Melhor caso: 22 minutos (234 clientes atendidos, tempo médio de 64 minutos)

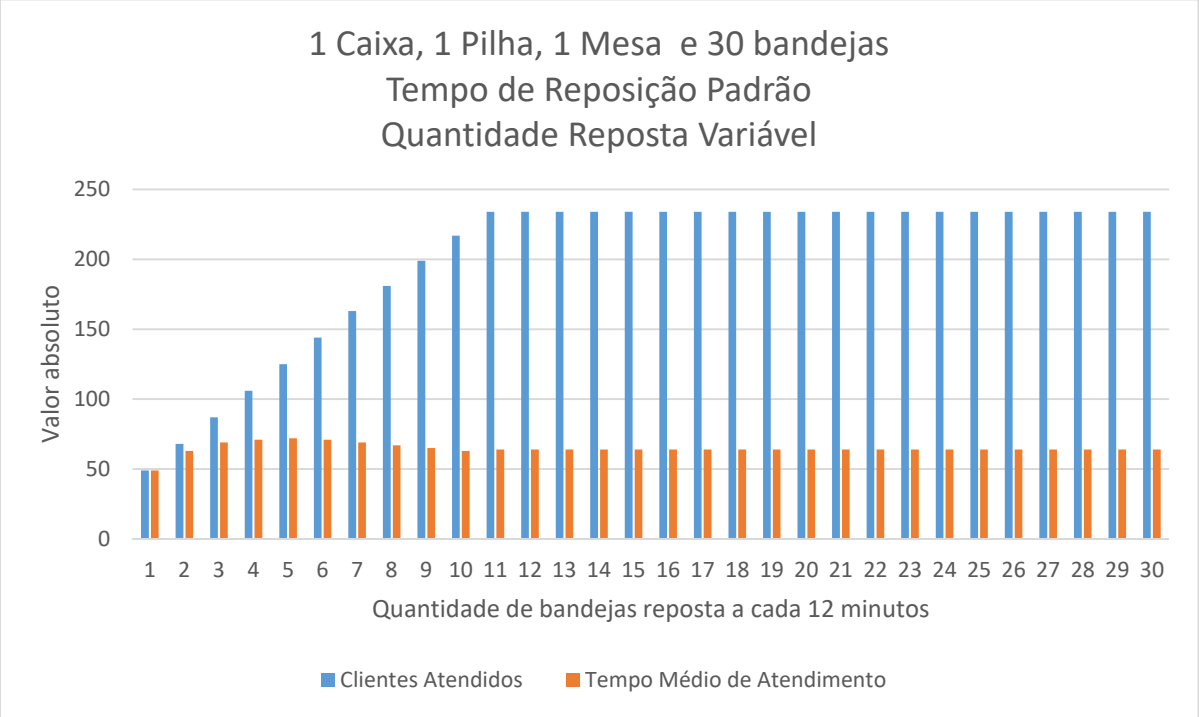


Melhor caso: 23 minutos (234 clientes atendidos, tempo médio de 64 minutos)

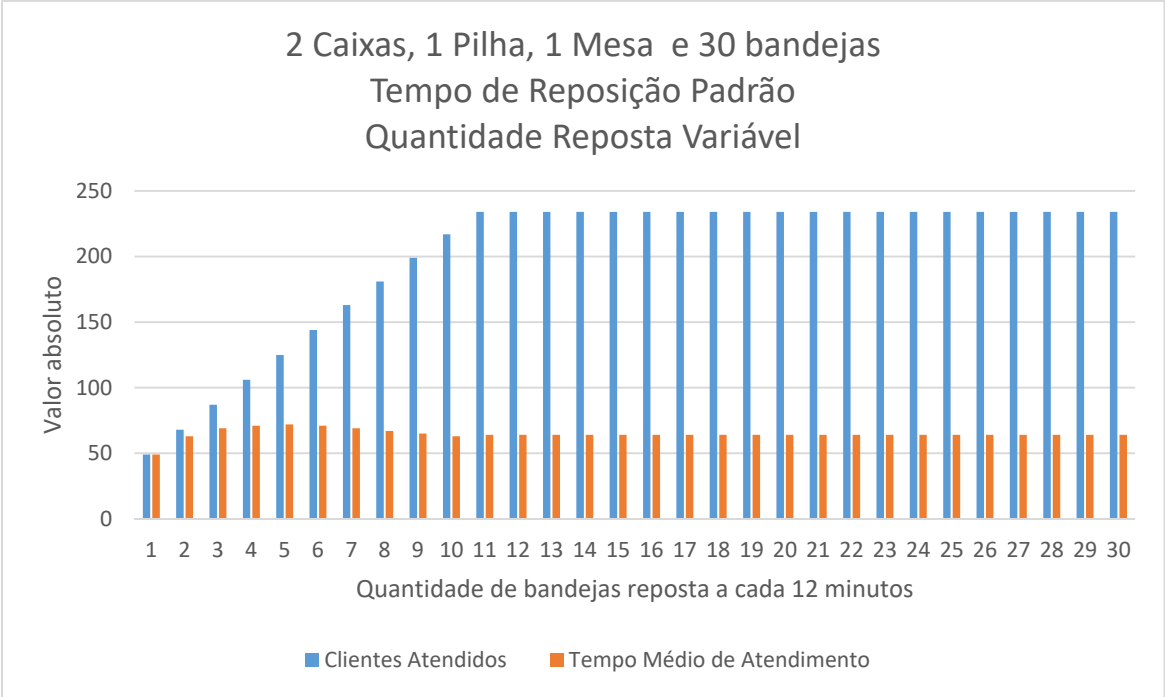


Melhor caso: 11 minutos (234 clientes atendidos, tempo médio de 64 minutos)

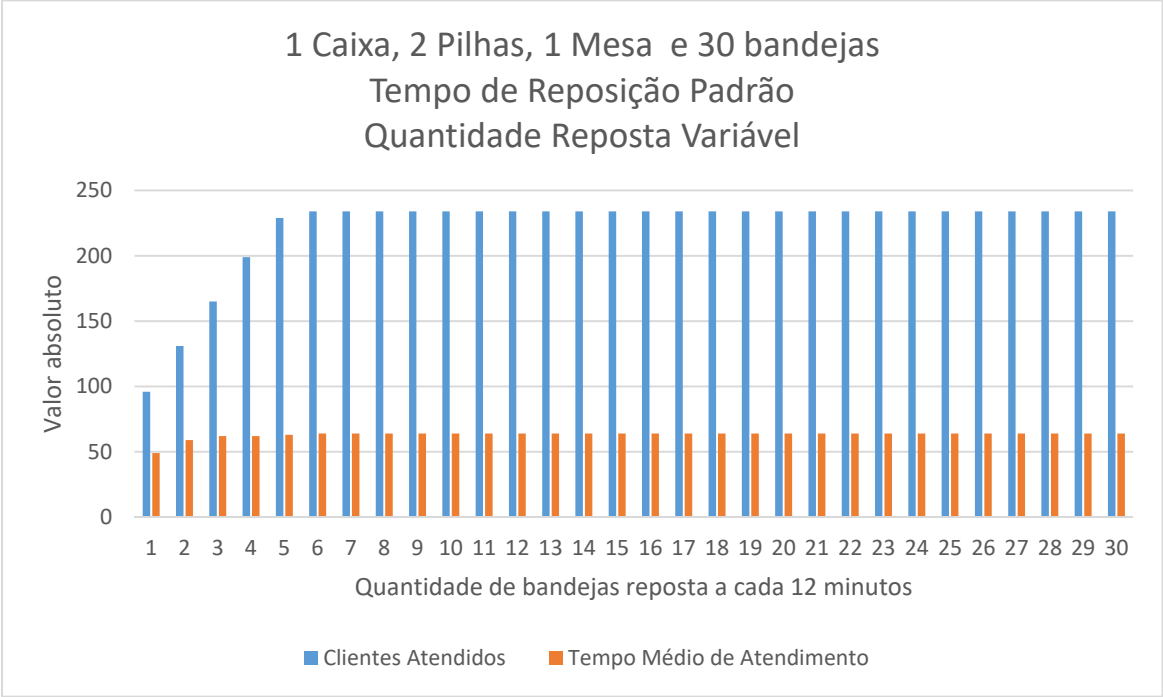
5.3 – Variando-se a quantidade de bandejas repostas



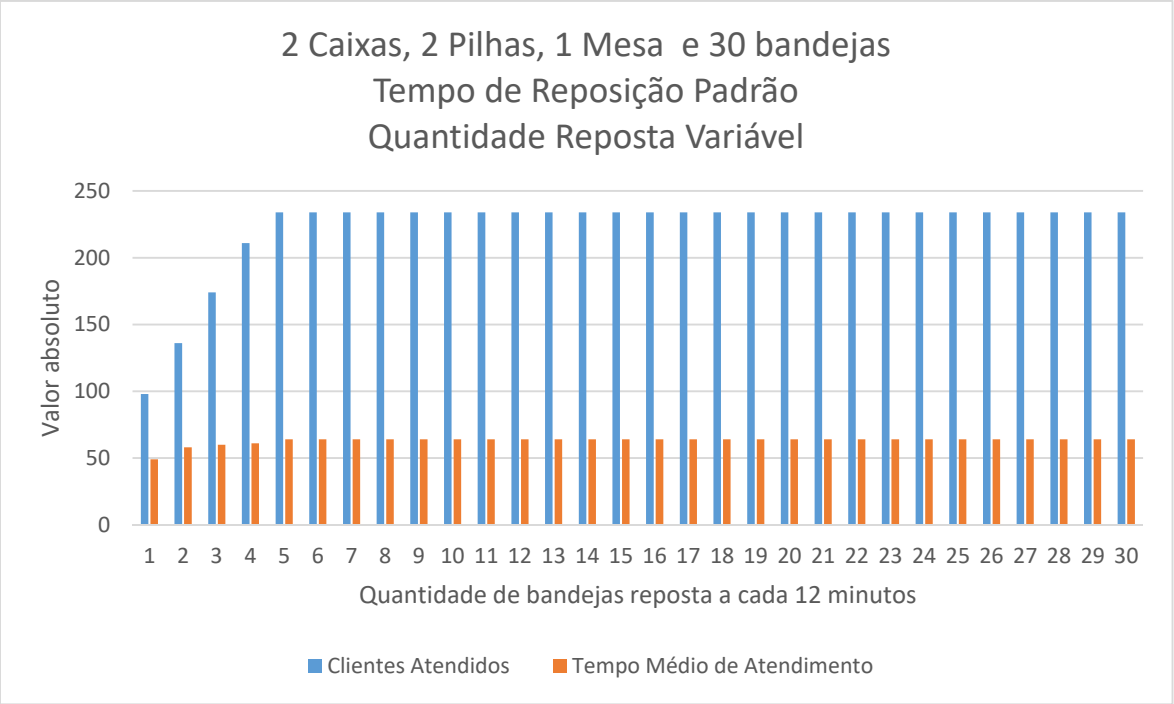
Melhor caso: 11 bandejas (234 clientes atendidos, tempo médio de 64 minutos)



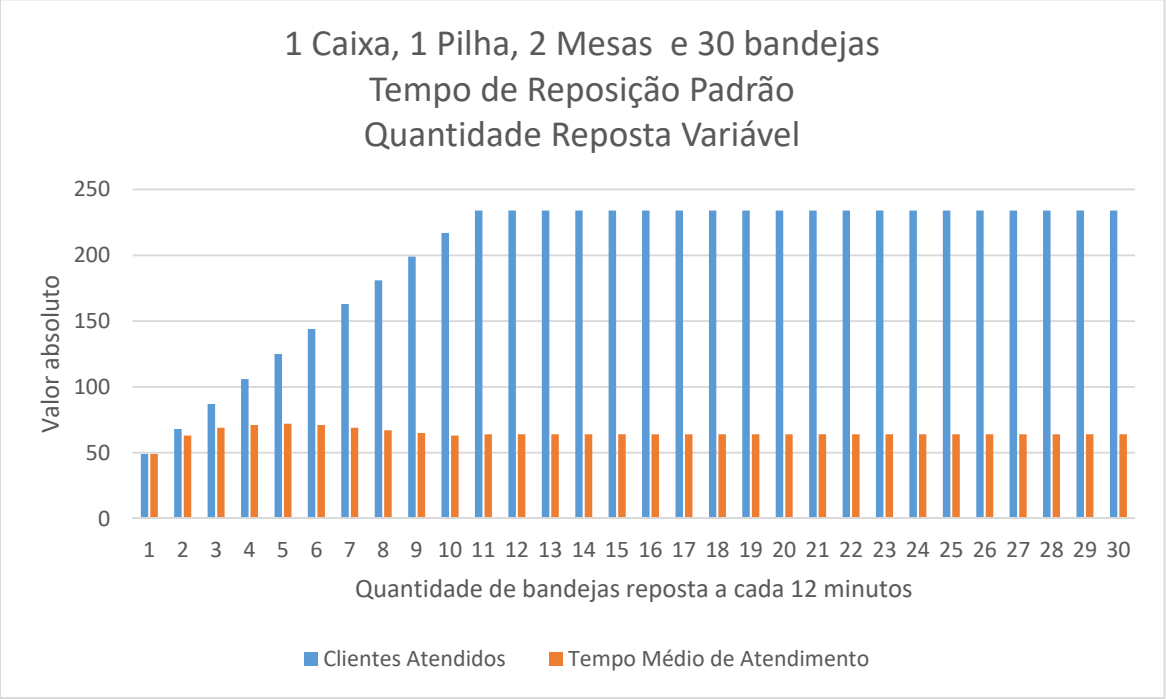
Melhor caso: 11 bandejas (234 clientes atendidos, tempo médio de 64 minutos)



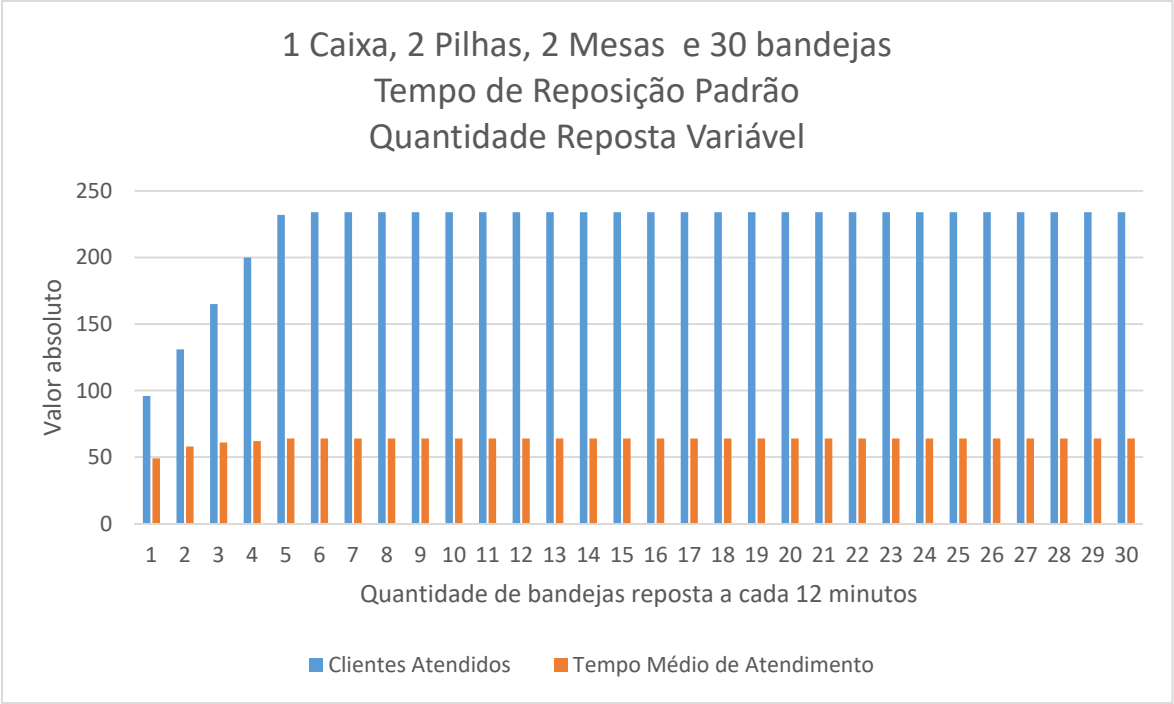
Melhor caso: 6 bandejas (234 clientes atendidos, tempo médio de 64 minutos)



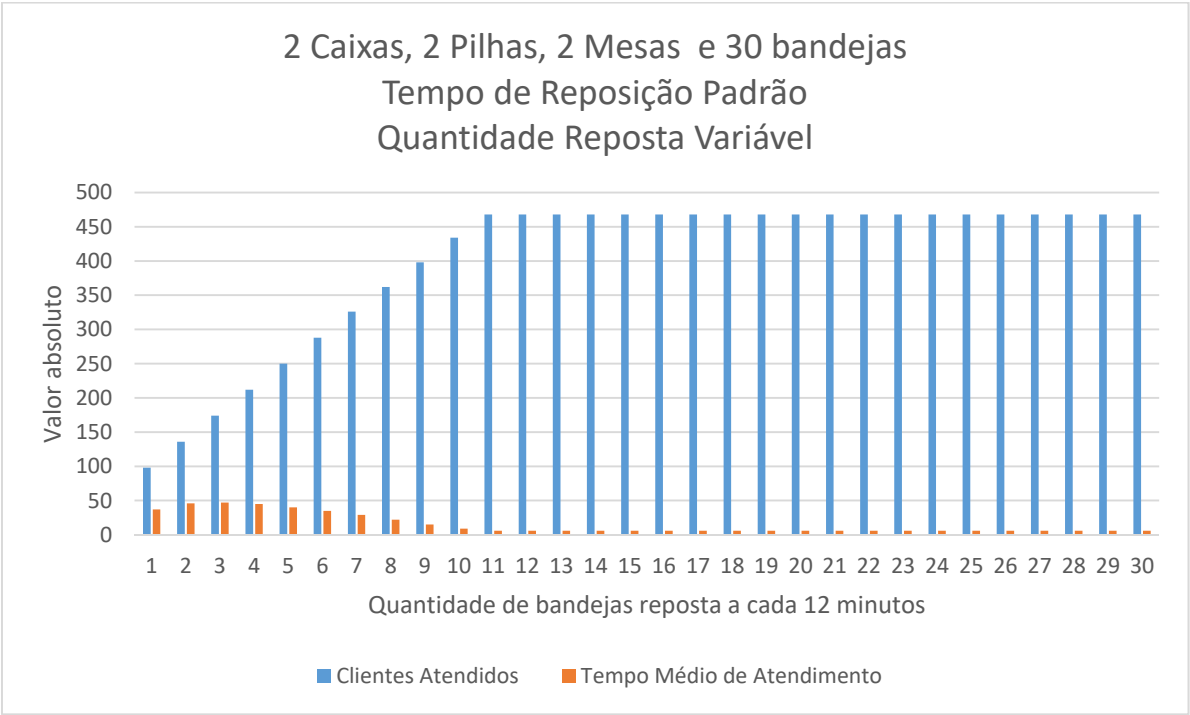
Melhor caso: 5 bandejas (234 clientes atendidos, tempo médio de 64 minutos)



Melhor caso: 11 bandejas (234 clientes atendidos, tempo médio de 64 minutos)



Melhor caso: 6 bandejas (234 clientes atendidos, tempo médio de 64 minutos)



Melhor caso: 11 bandejas (468 clientes atendidos, tempo médio de 6 minutos)

6 – Conclusão

Conforme os testes realizados, podemos observar que a eficiência do processo pode ser melhorada com pequenas alterações nos procedimentos.

Dentro da conformação estrutural do caso base (1 caixa, 1 pilha de bandejas e 1 mesa de alimentos), é possível aumentar em cerca de 8% o número de clientes atendidos, mantendo-se o tempo médio de atendimento praticamente inalterado, de, pelo menos, 3 formas diferentes: reduzindo-se o tempo para reposição das 10 bandejas em 1 minuto (de 12 minutos para 11 minutos), aumentando-se a quantidade de bandejas na pilha em 17 unidades (de 30 para 47) ou aumentando-se em uma unidade a quantidade de bandejas repostas a cada período de 12 minutos (de 10 para 11 bandejas).

Alterando-se a conformação estrutural, novos modelos podem se mostrar capazes de atender o maior número possível de clientes no período (considerando o mesmo fluxo de chegada de clientes), mantendo-se o tempo mínimo de 6 minutos para o atendimento de cada um dos clientes. Essas alterações consistem em dobrar o número de caixas, o número de pilhas e o número de mesas de alimentos. Nessa nova estrutura, 2 pilhas com 11 bandejas cada, ou um tempo de reposição de 11 minutos ou um número de bandejas repostas de 11 a cada 12 minutos seria suficiente para atender a todos os clientes no tempo mínimo.