Universidade Federal
do Rio de Janeiro
Escola Politécnica

DESENVOLVIMENTO DE INTERFACE WEB DE SIMULADOR DE PROCESSAMENTO MINERAL

Henrique Karl Fernandes Maia

Projeto de graduação apresentado ao Curso de Engenharia Metalúrgica da Escola Politécnica, Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Engenheiro.

Orientador: Rodrigo Magalhães de Carvalho

Rio de Janeiro

Setembro de 2018

DESENVOLVIMENTO DE INTERFACE WEB DE SIMULADOR DE PROCESSAMENTO MINERAL

Henrique Karl Fernandes Maia

PROJETO DE GRADUAÇÃO SUBMETIDO AO CORPO DOCENTE DO DEPARTAMENTO DE ENGENHARIA METALÚRGICA E DE MATERIAIS DA ESCOLA POLITÉCNICA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO DE MATERIAIS

Prof. Rodrigo Magalhães de Carvalho, D.Sc.

Prof. Luís Marcelo Marques Tavares, Ph.D.

Eng.ª Juliana Segura Salazar, M.Sc.

Rio de Janeiro – RJ, Brasil

Setembro de 2018

Karl Fernandes Maia, Henrique
Implementação de simulador aberto de circuitos de cominuição / Henrique Karl Fernandes Maia. -- Rio de Janeiro, 2018.
70 f. : il
Orientador: Rodrigo Magalhães de Carvalho.
TCC (Graduação - Engenharia de Materiais) - Universidade Federal do Rio de Janeiro, UFRJ/ESCOLA POLITÉCNICA, 2018.
1. Mineração. 2. Simulação. 3. Modelagem. 4. Cominuição. 5. LTMSim. I. Magalhães de Carvalho, Rodrigo. II. Implementação de simulador aberto de circuitos de cominuição.

AGRADECIMENTO

Aos meus pais Deise e Gilmar, por todo apoio dado em todos os meus anos de vida acadêmica. Foram eles que abriram meus olhos para o caminho acadêmico, desde o ensino técnico e agora à minha graduação. Além de tudo isso, ao apoio e suporte dados durante o desenvolvimento do projeto, a ajuda que me deram foi essencial para a conclusão desse projeto.

Ao professor Rodrigo, pela orientação, pela paciência e principalmente pela dedicação em me ajudar em todos os momento. Sua orientação foi essencial para que esse projeto fosse possível.

A Ana Beatriz, por todo apoio que me deu e por ser minha confidente em todos os meus momentos difíceis, você esteve comigo e me acompanhou quando eu demonstrei meu lado menos otimista e precisei de ajuda para voltar.

A equipe do Laboratório de Tecnologia Mineral, não só por toda ajuda e suporte, que foram perfeitos, mas também por me receber de braços abertos e por me ajudarem em todos os momentos, com todas as dúvidas e dificuldades com as quais cada um me ajudou. Fiquei muito feliz de fazer parte da equipe.

Aos meus amigos e colegas de curso, por compartilharem comigo todos os momentos da graduação. A convivência com tantas pessoas de tal nível de excelência certamente teve impacto significativo na minha formação profissional.

Ao corpo docente do departamento da Metalmat, pela contribuição na minha formação.

Aos meus companheiros de trabalho no BTG Pactual, em especial a Cristina Santos e ao Diego Borsato por terem compreendido e me incentivado a focar no projeto, mesmo em momentos de intensa demanda no banco.

Ao IFRJ/CefetQ/ETFQ, instituição que teve imenso impacto na minha formação acadêmica, profissional e pessoal. Muito do que sou hoje foi moldado lá e sou grato até hoje por tudo. Em especial a professora Janaína Nascimento, por todo o aprendizado e incentivo que me proporcionou.

Resumo do Projeto de Graduação apresentado à Escola Politécnica/UFRJ como parte dos requisitos necessários para obtenção do grau de Engenheiro de Materiais.

Implementação de simulador aberto de circuitos de cominuição

Henrique Karl Fernandes Maia

Setembro/2018

Orientador: Rodrigo Magalhães de Carvalho
Curso: Engenharia de Materiais

Ao trabalhar com beneficiamento mineral é essencial que seja feito um planejamento adequado de como será tratado o minério até que ele chegue a um produto final, de modo que um planejamento adequado é um dos grandes fatores que determina não só o potencial de rentabilidade da explotação de um minério, mas até mesmo a sua própria viabilidade. Para realizar esse planejamento, uma das ferramentas mais efetivas são os simuladores de processos, que são capazes de prever o comportamento de um sistema de beneficiamento sem que seja necessário o comprometimento com uma planta piloto. Ao trabalhar com pesquisa e desenvolvimento na indústria mineral, existe a necessidade constante de testes que muitas vezes envolvem modelos que ainda não foram publicados ou que foram publicados muito recentemente. Dessa forma, tais modelos ainda puderam ainda ser implementados dentro das soluções comerciais de simulação. A partir dessa necessidade, esse trabalho se propõe a demonstrar a implementação de um simulador aberto a novos modelos, que pode ser facilmente aprimorado de acordo com a necessidade do usuário.

Palavras-Chave: Mineração, Simulação, Modelagem, Cominuição, LTMSim

Abstract of Undergraduate Project presented to POLI/UFRJ as partial fulfillment of the requirements for the degree of Materials Engineer

Implementation of open simulation for comminution circuits

Henrique Karl Fernandes Maia

September/2018

Advisors: Rodrigo Magalhães de Carvalho

Course: Materials Engineering

When working with mineral processing, it is essential that the path between the run-of-mine and the final product has a very careful planning, as an adequate planning is one of the biggest factors to rule not only the potential profitability of exploiting an ore, but also it`s viability. To be able to do such planning, one of the most effective tools available are the process simulators, which are capable of predicting how a processing system is going to behave without the need for a pilot plant. When working with research and development for the mining industry, there is a constant need for testing that often involves models that were not yet published or that are too recent. Therefore, such models did not have the chance to be implemented into the commercial solutions for simulation. From this necessity, this work intends to demonstrate the implementation of a simulator that is open to new models, that can be easily be improved according to the user's needs.

Keywords: Mining, Simulation, Modeling, Comminution, LTMSim

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 – Mineração e beneficiamento mineral

A mineração é o processo de extração de minerais valiosos da crosta terrestre. Os processos de mineração tem papel importante na história da humanidade, andando em paralelo com o desenvolvimento da civilização, de modo que algumas das eras mais importantes da antiguidade são caracterizadas por diferentes tipos de minérios e os produtos de sua mineração - Era da Pedra, Era do Bronze, Era do Ferro, Era do Aço e Era Nuclear (HARTMAN e MUTMANSKY, 2002).

No processo de mineração, uma das etapas mais importantes é o beneficiamento do minério extraído visando obter um produto com características controladas. O beneficiamento mineral consiste no conjunto de operações que são aplicadas aos bens minerais visando modificar características do mesmo, como a granulometria, a concentração relativa das espécies presentes ou a sua forma, mas evitando alterar a identidade física e química dos minerais (LUZ *et al*., 2010).

Uma planta de processamento mineral tem um alto custo tanto de construção como de operação (GUPTA e YAN, 2016). Com isso, um planejamento efetivo e detalhado é imprescindível para a viabilidade econômica de um projeto dessa natureza, pois qualquer ganho de produtividade ou redução de custo do processo como um todo pode representar um grande impacto financeiro.

1.2 – Simulação como ferramenta de otimização

Ao pensar em projetar um sistema de processamento mineral percebe-se que há um significativo número de variáveis a serem levadas em conta, tanto em relação aos equipamentos disponíveis quanto em relação às características do minério a ser processado. São parâmetros importantes a serem conhecidos: a composição do minério,

sua densidade, a forma em que ele se encontra, a distribuição dos diversos componentes dentro do minério, impurezas contidas, minerais que podem ser extraídos e a homogeneidade do minério. A quantidade e complexidade dessas informações torna a tarefa de otimização de uma operação de beneficiamento de um dado minério praticamente impossível de ser realizada utilizando-se modelos simples.

Uma das formas que hoje se mostra muito eficiente nesse tipo de planejamento é a simulação de processos. Utilizando simulações computacionais é possível estimar o desempenho de diversas configurações de processos sem que seja necessário realizar e testar cada uma delas. Em uma simulação, cada operação unitária do processo é simulada usando um modelo numérico que visa prever tanto o perfil dos produtos de cada processo, por exemplo, granulometria, bem como o consumo energético desse processo. A partir do momento em que é possível comparar as diferentes configurações de circuito que podem ser usadas para organizar uma planta de beneficiamento mineral, é possível usar esse tipo de técnica para escolher a configuração que mais se aproxima de uma solução ótima para o problema.

A simulação de uma operação unitária é feita usando um modelo matemático apropriado, de modo que cada operação pode ser descrita por diversos modelos, que recebem diferentes parâmetros de entrada e estimam tanto o produto da operação unitária quanto outros parâmetros relevantes do processo. Tais simulações ainda podem ser em regime estacionário ou regime dinâmico. Simulações de regime estacionário representam o estado em que um fluxo de processos alcança estabilidade e funciona de forma constante, sem mudar de acordo com o tempo. Já as simulações de regime dinâmico representam estados em que esse equilíbrio ainda não foi alcançado, de modo que a cada momento o sistema apresenta um estado diferente. Simulações de regime estacionário representam bem o comportamento de um fluxo de processos após alcançar um estado de estabilidade. Já as simulações de regime dinâmico representam bem o comportamento do fluxo de processos antes que essa estabilidade seja alcançada, descrevendo por exemplo, o momento em que uma planta de processamento é iniciada até alcançar a estabilidade ou então o momento em que a planta é desligada.

1.3 – Desafios e lacunas

Existem hoje diferentes simuladores de processos na indústria mineral, cada um com diferentes pacotes de modelos matemáticos. Dentre os mais famosos existem, o USIMPAC®, que usa modelos propostos por Whiten (WHITEN, 1972), o JKSimMet®, que usa os modelos propostos por Lynch (LYNCH e NAPIER-MUNN, 1992), o MODSIM®, que usa os modelos propostos por King (FORD e KING, 1984) e mais recentemente, o Integrated Extraction Simulator (IES), que é um novo simulador com interface web e hospedado na nuvem, o que permite que o mesmo possa ser acessado de qualquer lugar com acesso à internet e um navegador. Esse último simulador tem uma variedade mais extensa de modelos. Além disso, os simuladores comerciais em geral trabalham constantemente para se manterem atualizados em relação a novos modelos que são desenvolvidos, aprimorando o acervo de modelos usados na simulação de processos.

Entretanto, ao trabalhar com pesquisa e desenvolvimento na indústria mineral, existe a necessidade constante de testes que muitas vezes envolvem modelos que ainda não foram publicados ou que foram publicados muito recentemente. Dessa forma, tais modelos ainda puderam ainda ser implementados dentro das soluções comerciais de simulação.

Em 2007, foi desenvolvido por Carvalho o LTMSim, com o propósito de simular o comportamento de circuitos de moagem. Essa versão inicial foi usada na dissertação de mestrado de Silva (2007) com o objetivo de simular duas usinas de moagem de escória para fabricação de cimento. Na época nenhum dos simuladores comerciais disponíveis era capaz de simular todas as operações unitárias envolvidas no processo descrito por Silva (2007), de modo que seriam necessários diversos simuladores diferentes para realizar as simulações necessárias. O LTMSim aproveitou a possibilidade de manipulação de diagrama de blocos oferecida pelo software Simulink (MathWorks, 1984) e utilizou funções escritas na linguagem do Matlab, como mostrado por Carvalho (CARVALHO, 2007), para realizar os cálculos de processo. As figuras 1.1 e 1.2 ilustram um dos fluxogramas estudados por Silva (2007), simulado usando o LTMSim:

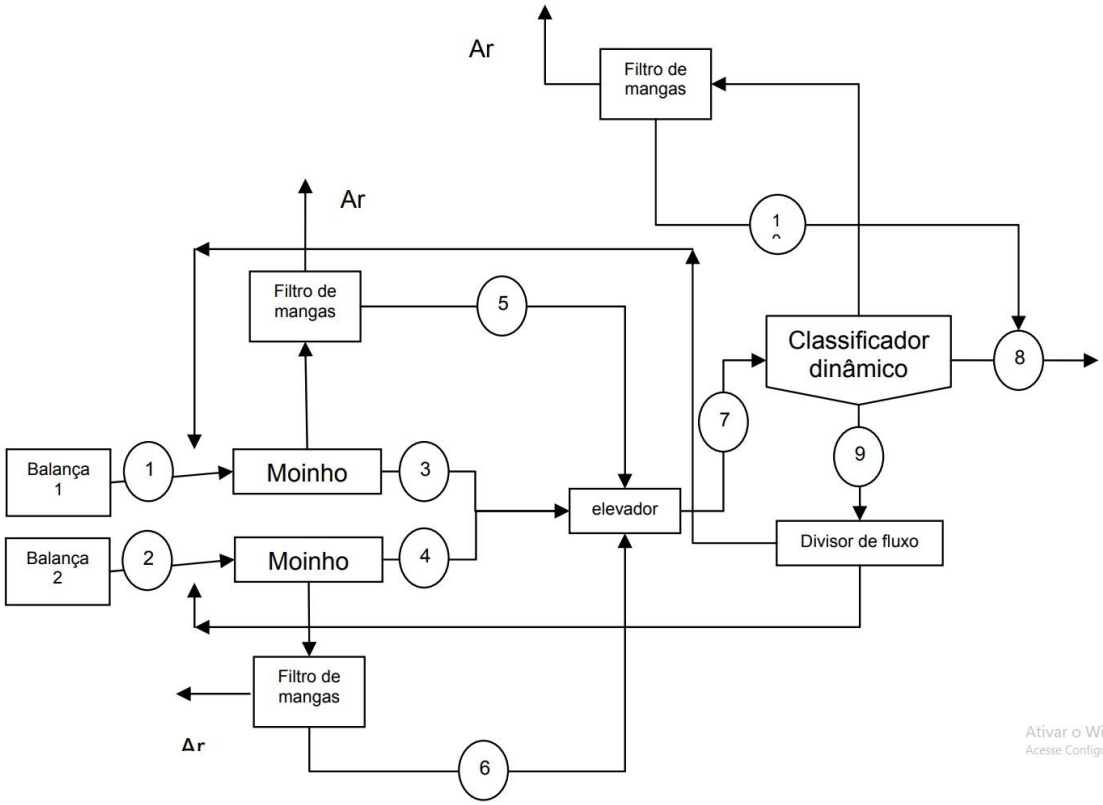

Figura 1.1 - Fluxograma de usina de moagem de escória (Silva, 2007)

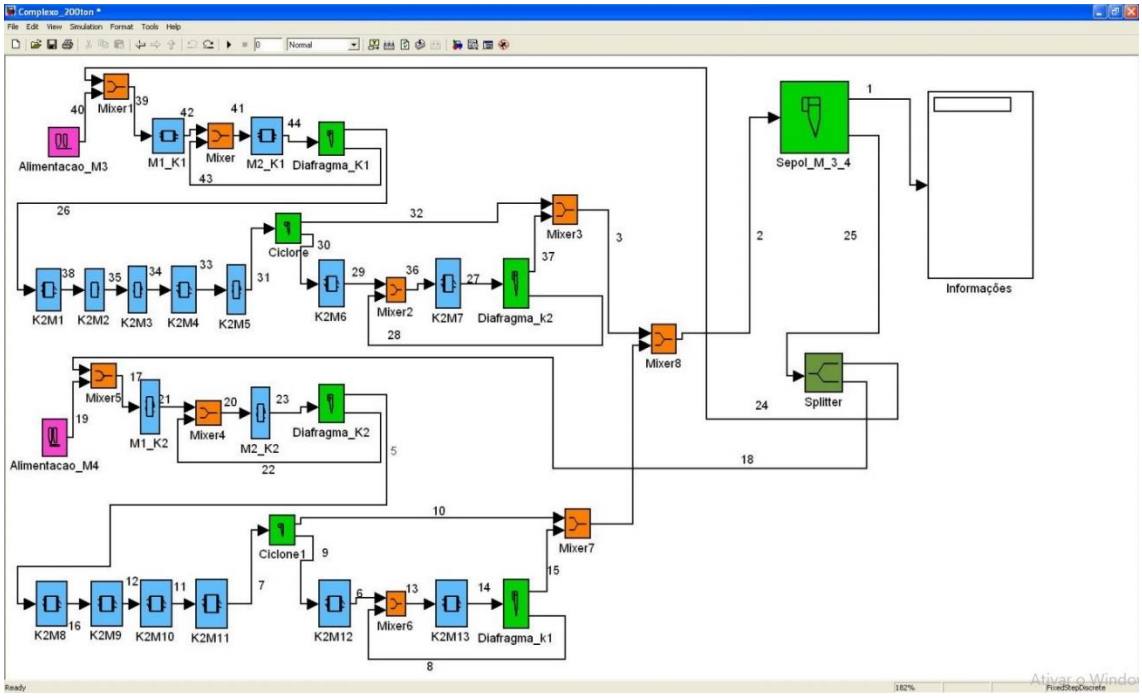

Figura 1.2 – Fluxograma de moagem de escória (Silva, 2007)

Os moinhos são representados por uma série de misturadores perfeitos com classificação interna por exaustão e grelha. Ambos os produtos dos moinhos são direcionados ao classificador dinâmico Sepol, cujo o fluxo de *oversize* repartido em 50% e cada fração direcionada a alimentação dos moinhos em paralelo.

Posteriormente, o LTMSim foi aprimorado a partir do trabalho de Alves (2011), que implementou um algoritmo de resolução sequencial no simulador de processos de beneficiamento de minérios, onde foi incorporado um algoritmo heurístico de otimização por localização de ciclos através da estratégia modular, reduzindo o número de iterações e, por conseguinte, o tempo de resolução. As metodologias usadas na programação do algoritmo também permitiram que o simulador usasse modelos propostos por diferentes autores, aumentando sua flexibilidade (ALVES, 2011).

1.4 – Objetivo

A partir do LTMSim, foi proposto o desenvolvimento de uma nova ferramenta de simulação, que seja aberta, de forma a importar diferentes tipos de modelos matemáticos de operações unitárias de beneficiamento mineral de acordo com a necessidade dos usuários, sem que o mesmo dependa do desenvolvimento da ferramenta em si, dando maior liberdade para que a comunidade possa desenvolver e testar a interação de novos e diferentes modelos dentro de fluxogramas complexos.

Desta forma, o presente trabalho trata do desenvolvimento de uma plataforma de simulação de processos de beneficiamento mineral com processamento remoto. Essa plataforma tem como escopo principal as seguintes características:

a) Capaz de simular fluxogramas simples ou complexos, podendo conter processos que sofrem retroalimentação (que são alimentados com o próprio produto);
b) Possibilidade de usar e cadastrar diferentes modelos matemáticos para realizar a simulação das operações unitárias propostas, tendo uma interface simples onde tais modelos podem ser cadastrados;
c) Será hospedado em um servidor, de modo que a ferramenta poderá ser acessada de qualquer computador com acesso à internet e um navegador.

Com essa nova ferramenta será possível testar diferentes configurações de plantas de processamento, usando diversos tipos diferentes de modelo, que ser

cadastrados facilmente, de modo a comportar rapidamente os novos modelos que vêm sendo desenvolvidos, mesmo antes que os mesmos sejam incorporados nos softwares comerciais.

Capítulo 2

Operações unitárias

As operações unitárias presentes em um circuito de beneficiamento mineral podem ser divididas em principalmente quatro tipos: transformação, separação, concentração e outras, sendo descritas em detalhe a seguir.

2.1 – Transformação física

Os processos de transformação são responsáveis por transformar o perfil da granulometria de um minério através da ação mecânica, reduzindo o tamanho das partículas presentes. A redução do tamanho das partículas é necessária tanto devido à separação dos minérios valiosos da ganga (quando se tem um material heterogêneo), quanto para facilitar a utilização do material em outros processos.

Normalmente, para que ocorra uma liberação satisfatória do mineral desejado, é necessário que o minério seja reduzido a uma granulometria fina. Dessa forma, a fragmentação se desenvolve por meio de três estágios: grossa, intermediária e fina. Os dois primeiros estágios (grossa e intermediária) são feitos usando britadores, enquanto o estágio de fragmentação fina é feito usando moinhos.

Em geral, os modelos que descrevem processos de cominuição se baseiam em entender a energia necessária para realizar a quebra de uma partícula em partículas menores e aplicar esse conceito para estimar o quanto as partículas de uma mistura serão reduzidas em um dado processo.

Baseado nisso, pode-se observar a equação 2.1, que descreve a relação entre a variação no tamanho de uma partícula e a energia necessária para alcançar essa variação:

$$\frac{dE}{dd_r} = f(d_r) \qquad \text{Equação 2.1}$$

Diversos autores propuseram diferentes formulações para $f(d_r)$, sendo as três mais comuns são as abordagens de Kick (1883), Rittinger (1857) e Bond (1952), onde cada uma sugere uma forma diferente da equação 2.2:

$$f(d_r) = -Kd_r^{-n} \quad \text{Equação 2.2}$$

Outra equação que é importante para a modelagem de processos de cominuição é a função de quebra, que descreve a quebra de uma partícula segundo uma função $B(x, y)$. A equação 2.6 mostra um possível formato para a função de quebra, descrita na equação 2.3 (AUSTIN e LUCKIE, 1972):

$$B(x,y) = K\left(\frac{x}{y}\right)^{n_1} + (1-K)\left(\frac{x}{y}\right)^{n_2} \quad \text{Equação 2.3}$$

onde K, n_1, n_2 são parâmetros da função quebra.

Finalmente, tem-se também a função de seleção, descrita pela Equação 2.4 (AUSTIN e LUCKIE, 1984):

$$\frac{dw_i(t)}{dt} = -S_i w_i(t) \quad \text{Equação 2.4}$$

Onde w_i é a fração de partículas pertencente a classe de tamanhos i e S_i é a taxa específica de quebra para esse tamanho.

Tendo as equações de quebra e seleção, pode-se descrever o processo de cominuição segundo o modelo proposto por AUSTIN *et al*. (1986) e AUSTIN *et al*. (1987), que descreve a quantidade de partículas no produto para cada uma das classes de tamanho presentes. Esse modelo segue a equação 2.5:

$$Fp_i = Ff_i - WS_i w_i + W\sum_{\substack{j=1 \\ i>1}}^{i-1} b_{ij} s_j w_j \quad \text{Equação 2.5}$$

Que também pode ser simplificado, de acordo com a equação 2.6:

$$p_i = f_i - \tau S_i w_i + \tau\sum_{\substack{j=1 \\ i>1}}^{i-1} b_{ij} s_j w_j \quad \text{Equação 2.6}$$

Onde p_i corresponde a fração de partículas da classe de tamanho i no produto. f_i corresponde a fração de partículas da classe de tamanho i na alimentação. w_i corresponde a fração de partículas da classe de tamanho i no hold-up. τ é o tempo de residência média. F é o fluxo de massa da alimentação e do produto. W é o hold-up no equipamento.

2.1.1 - Britagem

Britagem pode ser definida de forma genérica como o conjunto de operações que tem como objetivo fragmentar os blocos de minérios vindos da mina, levando-os a uma granulometria adequada para o devido processamento (ou até mesmo para que ele seja utilizado com algum fim). A britagem é um estágio composto por sucessivas etapas e

diversos equipamentos, visando reduzir o tamanho das partículas do material e também liberar os minerais valiosos de sua ganga.

A britagem pode ser aplicada a fragmentos de diferentes tamanhos, desde rochas de 10 milímetros até 1 metro, de modo que não existe um circuito padrão para britar diferentes tipos de minério. Devido a esse amplo alcance de tamanhos de partícula, os processos de britagem são divididos em categorias de britagem primária, secundária, terciária e quaternária (LUZ *et al*., 2010).

2.1.1.1 - HPGR

O HPGR (High Pressure Grinding Rolls) é um tipo de britador de alta eficiência, capaz de alcançar menor consumo de energia para uma dada relação de redução, quando comparado aos moinhos convencionais de bolas (LUZ *et al*., 2010). Ao final do trabalho, o simulador desenvolvido é demonstrado através da implementação de um modelo que descreve o comportamento do HPGR

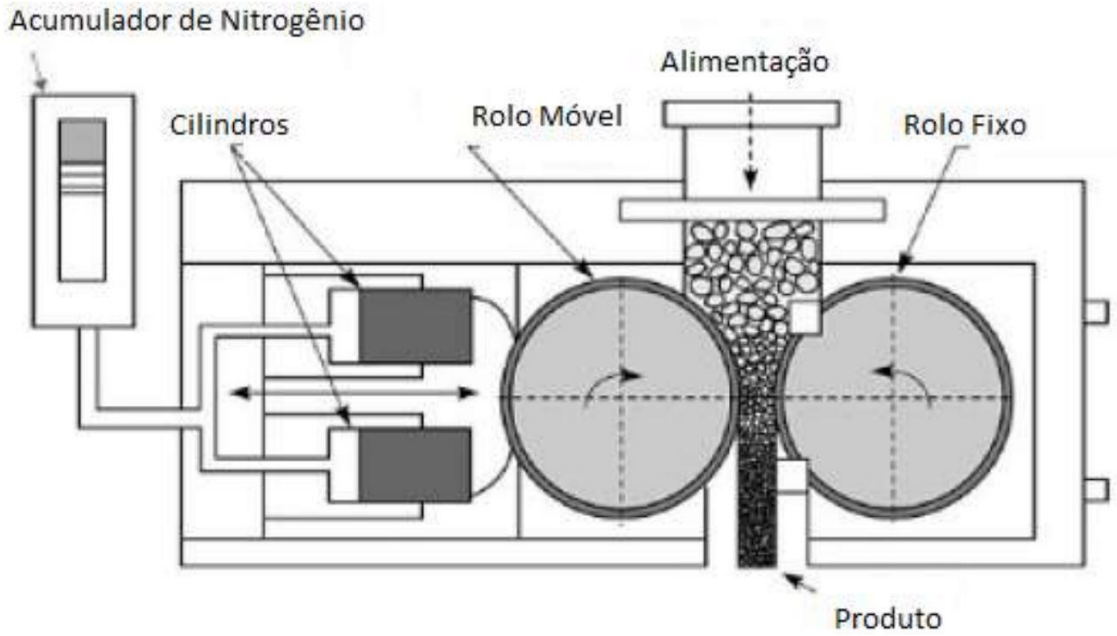

Figura 2.1 - Desenho esquemático da operação na HPGR (NAPIER–MUNN *et al*., 1996)

Existem diversos de modelos que descrevem o comportamento do HPGR, dentre eles tem-se o modelo proposto por TORRES e CASALI (2009). Esse modelo considera

a diferença na quebra das partículas de acordo com a região dos rolos pela qual cada partícula passou e com isso ele tem como resposta dois perfis granulométricos diferentes - aquele das partículas que passaram pelas bordas dos rolos e aquelas que passaram na região do meio dos rolos. Esse modelo é baseado nas funções de quebra e seleção (Equações 2.3 e 2.4):

Esse modelo foi recentemente abordado por CAMPOS (2018) visando levar em consideração a parcela do minério que transborda dos rolos, sem sofrer a ação de cominuição do equipamento. Essa consideração é feita a partir da implementação de um modelo empírico de capacidade que prevê a quantidade de material que sofre extrusão pela lateral do rolo:

$$Desvio = ae^{bx_{esp}}$$

Com isso, esse modelo tem diversos parâmetros, divididos em três classes:

a) Condições operacionais;
b) Parâmetros da função de quebra;
c) Parâmetros da função de seleção
d) Parâmetros do modelo de capacidade

O modelo proposto por Campos (2018) será usado na demonstração do simulador nos capítulos seguintes.

2.1.2 - Moagem

O último estágio dos processos de transformação ocorre em moinhos, onde as partículas são reduzidas pela combinação de impacto, compressão, abrasão e atrito, chegando ao tamanho necessário para que haja separação entre o mineral desejado e a ganga. Os processos de moagem são os mais complexos dentro da área de transformação. É a área que requer maior investimento, mais gasto de energia e é crítica em relação ao desempenho de uma planta de tratamento de minérios. Caso a moagem de um minério seja insuficiente, o resultado será um produto de granulometria grossa, com baixa liberação do mineral desejado, inviabilizando os processos de concentração. Já se a moagem for excessiva, o tamanho das partículas será reduzido em excesso de maneira

desnecessária, provocando maior consumo de energia e também perdas nos processos de concentração.

2.2 – Separação por tamanhos

Tendo em consideração a importância do controle da granulometria do minério a ser tratado, deve-se considerar também que esse minério será dividido conforme diferentes classes granulométricas, separando parcelas do produto que são mais grossas ou mais finas. A eficiência de um processo de classificação é descrita pela curva de partição, que representa a proporção de partículas presentes em cada tamanho da alimentação que é direcionada ao fluxo de partículas grossas, ou *oversize*. Além da eficiência, os processos de separação também apresentam um valor d_{50}, correspondente ao tamanho que tem 50% de chance de ir para o *overflow* e 50% de chance de ir para o *underflow*, correspondente ao tamanho de corte do processo (no caso de um peneiramento esse seria a abertura da peneira). São exemplos de processos de separação o peneiramento e o hidrociclone.

Os processos de separação podem ser previstos por modelos relativamente simples, baseados na curva de eficiência do processo, que a partir tamanho d_{50} do processo, estimam qual parcela de cada tamanho de partícula vai para o *underflow* ou para o *overflow*.

Além desse método, existem modelos mais robustos que tratam da resolução desse tipo de problema abordando-o através da CFD, *Computational Fluid Dynamics*, que (NARASIMHA, *et al.*, 2007), para prever o comportamento de classificadores que funcionam sob os princípios da fluidodinâmica, como por exemplo o hidrociclone e o classificador a ar.

2.3 – Concentração

Enquanto as operações de classificação, descritas na seção anterior são voltadas para a separação entre partículas de tamanhos diferentes, existem diversas outras características que podem ser apresentadas pelas partículas do minério, de modo que cada partícula apresenta uma distribuição dessas propriedades, o que diz respeito a concentração do mineral de interesse. A etapa de concentração é focada em usar essas

diferentes propriedades para separar os materiais de acordo com suas diferentes naturezas, ou seja, separar os minerais que são de interesse dos que não são. A partir do momento em que os minerais de interesse já não estão mais agregados ao resto do minério (após passarem pelas etapas de fragmentação e classificação). O propósito do processo de concentração, como diz o nome, é gerar um produto com maior concentração do mineral desejado em comparação com a mistura inicial. A separação entre minerais de diferentes naturezas é feita a partir de uma diferença física ou físico-química entre o mineral de interesse e os demais. A diversidade de características físicas e físico-químicas entre os minérios origina diferentes famílias de equipamentos de concentração, como por exemplo, a concentração gravimétrica, faz a separação através da diferença de densidade entre os diferentes materiais que compõem o minério.

Como os processos de concentração se baseiam em diferentes propriedades dos materiais separados, cada tipo de processo é baseado em um princípio diferente, de modo que os modelos que descrevem cada tipo de processo de concentração são completamente diferentes uns dos outros.

2.3.1 - Concentração Gravimétrica

A concentração gravimétrica é o processo que separa as partículas de uma mistura de acordo com as densidades, tamanho e forma das mesmas. Nesse caso, a separação é feita pela própria força da gravidade ou por forças centrífugas e utilizado um meio fluido (água ou ar) para efetivar a separação/concentração. Tradicionalmente esse tipo de separação é feita usando jigues, mesas vibratórias, espirais, cones e calhas (LUZ, 2010). Dada a variedade de diferentes equipamentos usados na separação gravimétrica, também existe uma diversidade de modelos usados para descrever o comportamento de cada tipo de equipamento.

2.3.2 – Separação em meio denso

Outro tipo de separação que é importante na indústria mineral é a separação em meio denso. Esse tipo de separação se diferencia, como o nome sugere, pelo meio em que as partículas estão suspensas em um meio com determinada densidade. Nos processos de concentração gravimétrica o meio denso utilizado normalmente é o ar ou a

água, que têm densidades muito baixas quando comparadas com as partículas de um minério, então todas as partículas presentes na dispersão são mais densas que o próprio meio. Na separação por meio denso, o meio em que as partículas estão dispersas possui densidade controlada intermediária à dos minerais que o processo se propõe a separar (DUTRA, 2008). Tendo um meio de densidade intermediária, o resultado desse tipo de separação é composto por uma parte que flutua (minerais com densidade menor que a do meio) e uma parte que afunda (minerais com densidade maior que a do meio) (ARAÚJO, 2012).

2.3.3 – Separador magnético

A separação magnética é usada na separação de minerais de acordo com a propriedade de suscetibilidade magnética, que descreve como um material responde a um campo magnético. A complexidade de modelar esse tipo de equipamento se dá pelo fato de que as partículas que estão sendo separadas contém diferentes teores de material ferromagnético. Com isso, não se trata de separar partículas que são atraídas ou não pelo campo magnético, mas de separar essas partículas de acordo com o quanto as mesmas são atraídas pelo campo. Existem diversos modelos tanto empíricos quanto fenomenológicos que descrevem o comportamento de um separador magnético (GONZAGA, 2014).

Dentre os modelos empíricos são notáveis os de DOBBY e FINCH (1977), que mostra que a recuperação das partículas depende não só da susceptibilidade magnética, mas também do tamanho da partícula.

O modelo empírico de TUCKER (1994), que é baseado em um modelo de concentração gravimétrica, onde a separação é representada por um conjunto de coeficientes de transferência. Esses coeficientes de transferência descritos como funções da intensidade do campo aplicado para cada tamanho e susceptibilidade magnética de partícula.

King também propôs um modelo empírico (Manual Modsim, 2001) baseado na função de Rosin-Rammler para representação da curva de partição. Esse modelo propõe que a separação é controlada pela composição volumétrica da partícula.

2.4 – Outras operações

Existem outras operações unitárias que não se encaixam nos tipos citados acima, mas que são importantes dentro do escopo deste trabalho:

a) Alimentação - consiste em toda entrada de minério dentro do sistema de beneficiamento mineral;
b) Pilha - consiste nos últimos elementos de um sistema de beneficiamento mineral, tendo assim uma ou mais pilhas de pó com maior pureza e menor distribuição granulométrica em relação aos fluxos de alimentação. Vale ressaltar também que desconsiderando perdas, a massa total das pilhas de pó equivale à massa total de minério nos fluxos de alimentação;
c) Mistura de fluxos - consiste em um processo com entrada de diferentes correntes que são misturadas em apenas uma corrente de saída;
d) Divisor de fluxo - atua como um divisor de fluxo dado um percentual de atuação definido. Pode se citar como exemplo, quando um produto do circuito de britagem é direcionado a diversos moinhos de bolas operando em paralelo.

Capítulo 3

Simuladores de Processo na Indústria Mineral

3.1 – Algoritmos

Ao se propor a estudar um simulador de processos, existem questões referentes ao escopo do trabalho que precisam ser atendidas. A primeira é a necessidade de utilização de um algoritmo de resolução sequencial para resolver os sistemas montados. A partir do momento em que se tem um sistema que representa uma planta de beneficiamento mineral, é necessário que também seja organizada a forma mais eficiente de alcançar a resolução do problema, ou seja, deve-se encontrar a sequência ótima para resolver cada um dos processos até chegar na solução. Além disso, a partir do momento em que o sistema foi simulado, deve-se definir o critério usado para decidir se a solução obtida é é aceitável para ser tida como resultado final da simulação, ou se ainda é necessário efetuar mais rodadas de simulação.

3.1.1 – Ciclor

Ao simular um conjunto de processos, esse conjunto pode formar uma estrutura cíclica ou acíclica. No caso de uma estrutura acíclica, a resolução do sistema através de simulação é simples, consistindo em simular cada operação unitária até que todas as operações sejam simuladas. Entretanto, no caso de estruturas cíclicas, essa resolução se torna mais complexa, pois nesse caso existe a formação de dependências circulares, que impossibilita a resolução de forma simples como é feita no caso de estruturas acíclicas, de modo que a solução só pode ser obtida por tentativas através da adoção de um procedimento iterativo. No entanto, a simulação de processos envolve estruturas complexas com inúmeros trechos cíclicos e acíclicos interligados, justificando o

estabelecimento prévio de uma estratégia de cálculo para a resolução (PERLINGEIRO, 2005).

Para resolver esse tipo de problema, foi desenvolvido o Ciclor (PERLINGEIRO, 2005), um algoritmo essencial para o funcionamento desse projeto, capaz de escolher quais correntes dentro de um sistema cíclico devem ser alimentadas (com quaisquer valores) para possibilitar a simulação (ALVES, 2011).

A escolha das correntes a serem alimentadas é feita seguindo o critério de quais correntes geram mais dependências dentro do processo como um todo, de modo que ele escolhe o menor número de correntes a serem alimentadas para que a simulação alcance uma convergência (PERLINGEIRO, 2005).

A partir disso é possível alimentar as correntes obtidas com valores arbitrários e então efetuar rodadas de simulação até que os resultados alcancem o critério de convergência escolhido.

3.1.2 – Critério de convergência

Com o resultado do Ciclor, é possível escolher correntes iniciais para serem alimentadas, possibilitando que todas as etapas do sistema sejam devidamente simuladas. Entretanto, os valores alimentados nas correntes escolhidas pelo Ciclor são arbitrários e não tem nenhuma relação com a realidade. Com isso, ao realizar sucessivas rodadas de simulação, cada rodada de simulação nos traz uma estimativa cada vez melhor do conjunto de correntes e resultados obtidos do processamento, convergindo para o valor que os mesmos teriam caso o sistema fosse executado na vida real.

Como as correntes e resultados tendem a convergir para um resultado, é preciso definir um critério para definir se os valores obtidos em uma rodada são aceitáveis para serem tratados como resultado da simulação. Um possível critério que pode ser usado é quando a diferença entre os resultados de sucessivas rodadas de simulação se torna suficientemente pequena (SLATER, 2008). Entretanto, surge a questão de como realizar esse tipo de julgamento quando a quantidade de variáveis verificadas não é definida, de modo que é necessário obter um indicador que possa ser usado como parâmetro para essa decisão. Para resolver esse problema existem diversas alternativas de modo que cada uma tem suas vantagens e desvantagens, e pode-se listar três das mais simples:

a) Uma opção é verificar o somatório dos erros de cada resultado, entretanto é fácil perceber que esse tipo de abordagem colocaria um viés sobre o critério de convergência de acordo com a complexidade do sistema a ser simulado, de modo que sistemas muito complexos tendem a ter mais dificuldade em atender esse critério de convergência;
b) Outra opção é verificar o erro médio dos parâmetros, que é igual ao somatório dos erros dividido pelo número de variáveis avaliadas, o que a primeira vista parece uma boa opção, mas ao adotar esse tipo de abordagem em um sistema ele sofrerá também com um viés de acordo com a complexidade do sistema, de modo que sistemas muito complexos são capazes de alcançar o critério de convergência mesmo quando nem todos os parâmetros estão convergindo, já que a média desses erros pode diluir um erro elevado em um dado resultado, fazendo com que ele passe despercebido;
c) Uma terceira opção para esse tipo de problema é tratar o erro máximo do sistema, de modo que o critério de convergência será o erro do resultado que apresenta o maior erro dentre todos os resultados de todos os processos. Essa abordagem apesar de criar um critério de convergência que em geral é mais difícil de ser atendido (já que a convergência será nivelada pelo resultado que apresenta maior dificuldade de convergir), traz uma forma de verificar a convergência do sistema sem sofrer com o viés gerado pela complexidade do sistema ou pela grande quantidade de variáveis que estão sendo observadas.

Além desses três critérios de convergência citados acima, existem diversos outros que são capazes de alcançar a convergência mais rápido, conforme descrito por FORD e KING (1984), são eles os métodos *bounded Wegstein*, (WEGSTEIN, 1958), *dominant eigenvalue* - autovalor dominante (ORBACH e CROWE, 1971), dentre outros métodos baseados na abordagem newtoniana (BARNES, 1965; BROYDEN, 1965; ROSEN, 1967). Entretanto, por mais que esses métodos apresentem uma convergência mais rápida comparados ao critério previamente proposto, a convergência desses métodos depende de que seja feita uma boa estimativa inicial para que seja obtida a solução do problema.

3.2 – Simuladores de processamento mineral em regime estacionário

Atualmente existem diversos *softwares* que podem ser usados para realizar a simulação de um sistema de beneficiamento mineral, cada um com suas próprias características. Observando as características é possível perceber que os diferentes simuladores presentes no mercado se complementam, de modo que no momento nenhum deles consegue superar todos os outros de forma plena. A tabela 3.1 mostra uma comparação das características de alguns dos simuladores mais adotados na simulação de processamento mineral:

Tabela 3.1 - Comparação entre diversos simuladores de processamento mineral

	MODSIM	USIM PAC	JKSimMet	IES
Acesso livre	Sim*	Não	Não	Sim**
Balanço de massa	Não	Sim	Sim	Não***
Calibração	Não	Não	Sim	Não***
Simulações automatizadas	Não	Sim	Não	Sim
Prototipagem	Sim	Não	Não	Sim
Otimização de parâmetros	Não	Não	Não	Sim

(*) - Versão Demo disponibilizada

(**) - Acesso livre para participantes do *CRC ORE 2* e patrocinadores e pesquisadores do projeto AMIRA P9Q

(***) - Funcionalidade não está presente, mas está dentro do escopo previsto para a versão final

Capítulo 4

Metodologia

4.1 – Arquitetura

Para a implementação do simulador foi escolhida uma arquitetura com duas camadas. A primeira camada é a de interface com o usuário, enquanto a segunda camada é responsável pelo acesso aos dados, pela lógica de simulação e pela interface com o Matlab. A arquitetura da aplicação é descrita na Figura 4.1:

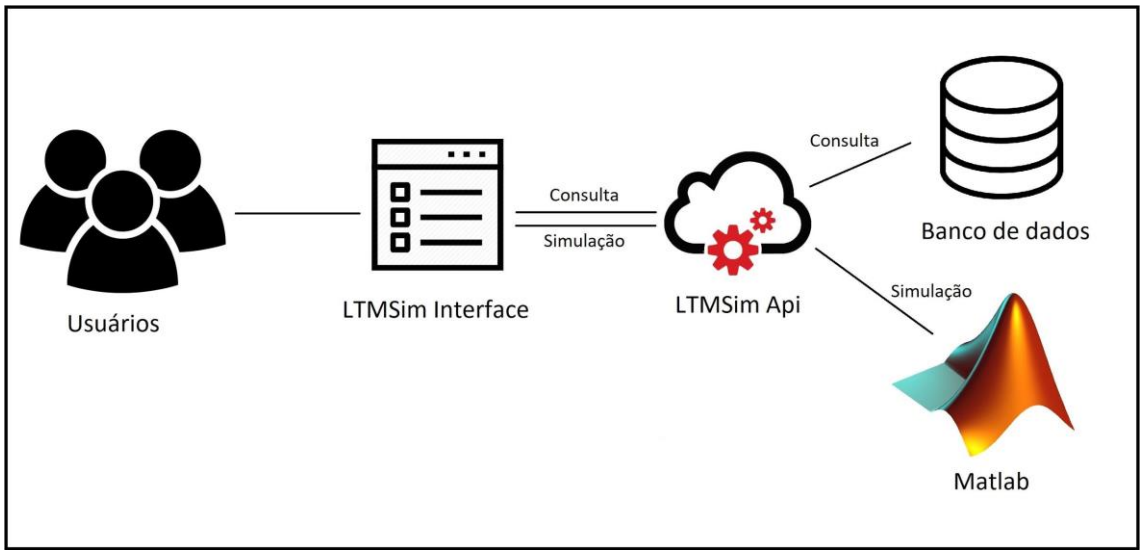

Figura 4.1 - Arquitetura do LTMSim Open

A camada de interface é acessada através de uma página na *Web* usando um navegador. Essa interface por sua vez se comunica com a camada de serviço (LTMSim Api) sempre solicitando alguma informação do banco de dados, enviando uma informação nova para o banco de dados ou então solicitando uma simulação de um fluxo de processos. Enquanto isso, a camada de serviço é capaz de processar e entregar para a interface todas as informações solicitadas.

Um serviço é uma aplicação capaz de expor suas próprias funções de modo que elas possam ser chamadas através de uma chamada de rede. Com isso, é possível expor,

por exemplo, uma função que busca no banco de dados quais são os processos cadastrados no sistema. Esse tipo de comportamento é o que permite que a interface consiga acessar a lógica e as informações necessárias para o funcionamento do simulador.

Essa arquitetura foi escolhida porque ela permite segregar as responsabilidades de apresentação do simulador, no caso a interface que é vista pelo usuário, das responsabilidades lógicas do mesmo, no caso a camada de serviço, que tem toda a inteligência da aplicação.

A partir dessa segregação, tem-se duas aplicações que funcionam de forma independente e com isso podem ser aprimoradas ou substituídas sem impactar no funcionamento do resto do sistema. Por exemplo, se em algum momento for percebido que uma interface *Web* não é a melhor forma de exibir o simulador para o usuário, pode-se criar uma interface completamente nova em qualquer outra linguagem sem precisar alterar a parte de lógica do simulador, que está contida na camada de serviço. Além disso, o inverso também ocorre, ou seja, é possível alterar e substituir a parte lógica do simulador sem causar nenhum impacto na interface que apresenta o simulador para o usuário.

Outro ponto positivo desse tipo de arquitetura é que, ao trabalhar no desenvolvimento de qualquer um dos componentes do projeto, não há risco de impacto no outro componente, já que eles funcionam de forma independente.

Além disso tudo, é importante justificar porque não foi adotada uma arquitetura mais robusta, segregando ainda mais as responsabilidades da aplicação. Segundo a metodologia adotada, um único serviço trabalha com três responsabilidades distintas: lógica de simulação, acesso a dados e interface com o Matlab. Esse excesso de responsabilidades em uma única aplicação é contrário ao conceito de arquitetura baseada em microserviços, pois o mesmo se baseia na divisão das responsabilidades entre diversos serviços em que cada um deles trabalha de forma independente dos outros e tem apenas uma responsabilidade (MICROSOFT, 2018).

Apesar disso, foi decidido trabalhar apenas com um serviço porque no estágio atual do simulador a aplicação ainda é relativamente simples, então a concentração da lógica em apenas um serviço foi um fator que facilitou o desenvolvimento da aplicação nessa fase inicial.

Outro motivo relevante para essa decisão é que sempre que ocorre uma comunicação com um serviço (seja entre dois serviços ou entre uma interface e um serviço), essa comunicação toma um tempo elevado. Isso acontece porque quando um método é chamado dentro de um código, o tempo de execução é baseado apenas na quantidade de processamento necessária do processador, enquanto um método acessado através da chamada de um serviço ainda precisa passar por uma chamada de rede até chegar ao seu destino (JUNIOR, 2015).

O tempo dessa comunicação é pouco perceptível em casos em que há poucas comunicações dessa natureza, por exemplo, quando a interface solicita uma informação do servidor, tratamos apenas de uma comunicação. Entretanto, em lógicas mais complexas, como na simulação de um fluxograma, se cada chamada feita no Matlab precisasse passar por uma interface de serviço, o tempo de execução de uma simulação ficaria muito mais alto quando comparado com a mesma simulação sendo feita por um serviço que agrega as duas responsabilidades (de organizar a lógica da simulação e executar os scripts de Matlab).

Normalmente esse problema de performance é resolvido através de várias chamadas em paralelo de um serviço a outro, de modo que cada chamada é feita antes que a chamada anterior seja finalizada. Entretanto, ao realizar a simulação de um fluxograma, nem sempre é possível realizar as chamadas em paralelo, já que para simular cada operação unitária, ela depende das correntes que ela irá receber de outros processos, impedindo o paralelismo

4.2 – Front-end

Para a camada de interface com o usuário (front end) foi escolhida uma interface web, que pode ser acessada através de um navegador de internet, dispensando qualquer instalação de software por parte do usuário final, além de ser compatível com computadores de diferentes sistemas operacionais. Foi escolhido para o desenvolvimento dessa camada o framework Angular 5, que vem com diversas ferramentas que facilitam e aceleram o desenvolvimento da aplicação. Uma aplicação web funciona a partir de três linguagens diferentes:

a) HTML, define o que o usuário vê;
b) CSS, define como é exibido cada elemento declarado no HTML;

c) JavaScript, define a lógica por trás dos componentes da tela.

4.2.1 – HTML

HTML significa HyperText Markup Language e é a linguagem padrão na criação de páginas e aplicações web (W3C). É nessa linguagem que são definidos os elementos de uma página web. Por exemplo, um parágrafo de texto é declarado da seguinte forma (Figura 4.2):

```
<!DOCTYPE html>
<html>
    <head>
          <meta charset="utf-8">
        <title>Titulo!</title>
    </head>
    <body>
          <h1>Titulo principal</h1>
          <p>Hello LTM!</p>
          <h2>Projeto LTMSim Open</h2>
          <p>HTML simples</p>
    </body>
</html>
```

Figura 4.2 - Exemplo de trecho de código HTML

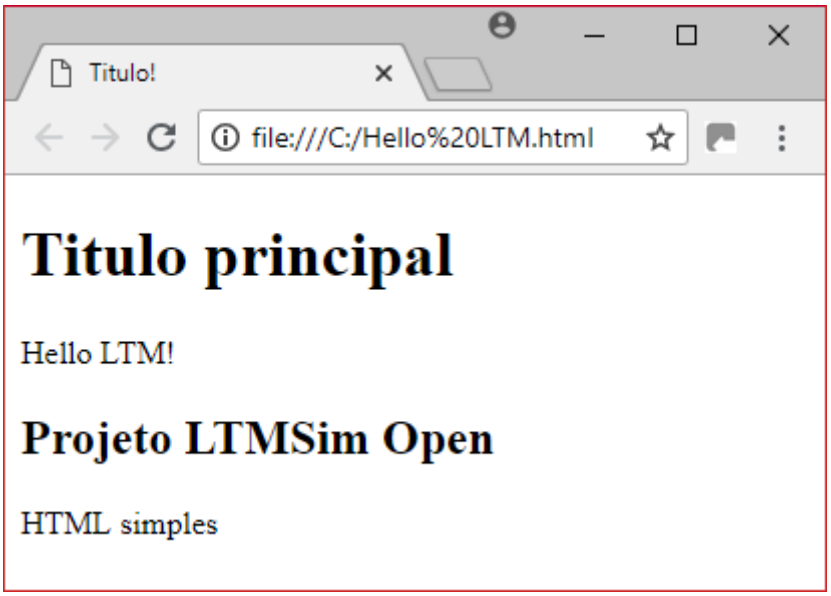

Figura 4.3 – Hello World de um documento em HTML

Combinando os elementos básicos de HTML é possível criar páginas ricas em conteúdo, com texto, tabelas, imagens e formulários.

4.2.2 – CSS

Como pode ser observado, a página feita no exemplo anterior contém somente algumas linhas de texto com formatação simples. Usando a linguagem CSS - Cascading Style Sheets - é possível descrever como cada elemento da página é exibido. Isso significa que o CSS é usado para definir fonte de texto, espaçamento entre os elementos, planos de fundo e até mesmo animações (como um botão sendo apertado). É usando a combinação entre HTML e CSS que o conteúdo dos sites modernos são exibidos. O CSS foi desenvolvido para permitir a separação entre apresentação e o conteúdo de uma página, provendo melhor flexibilidade e controle para a apresentação do conteúdo das páginas, de forma que diversas páginas HTML podem compartilhar um mesmo arquivo CSS (permitindo que as várias páginas de um site compartilhem a mesma formatação sem que o código referente a essa formatação seja repetido em cada página).

4.2.3 – JavaScript

4.2.3.1 - Básico

JavaScript é a linguagem de programação que é responsável por toda lógica por trás de uma página web (FLANAGAN, 2006). Enquanto o HTML e o CSS definem o que é mostrado em uma página e como esse conteúdo é apresentado, o JavaScript é responsável pela lógica da tela. É o JavaScript por exemplo que entende o que é um processo e como os processos de um fluxograma se relacionam.

Como é esperado de uma linguagem de programação de alto nível, o JavaScript é capaz de executar operações com números, strings e , eores além de tudo isso, ele é capaz de interagir com os elementos contidos no HTML da página.

Com isso, esse último elemento é responsável pela interatividade da página, de modo que é ele quem realiza consultas no servidor, retornando elementos requisitados. Um exemplo dessa atuação é por exemplo, quando um usuário acessa a tela de cadastro de um modelo de processo.

Para que a tela exiba de forma correta as informações do modelo, é necessário que o código em JavaScript faça a requisição dessas informações para o servidor. Com essas informações é possível exibir na tela o nome do modelo, de quais parâmetros ele depende e que tipo de resposta é esperada dele ao término da simulação.

Entretanto, existem tarefas que são muito comuns no desenvolvimento de páginas web, mas que ainda são relativamente trabalhosas e complexas de serem implementadas em JavaScript. Por isso, a comunidade de usuários da linguagem desenvolve constantemente ferramentas que facilitam a implementação dessas tarefas e que provém essas implementações seguindo as melhores práticas disponíveis.

4.2.3.2 - TypeScript

Além do JavaScript existe também a linguagem TypeScript, que é um superset do JavaScript, onde todas as funcionalidades do JavaScript podem ser usadas em um código em TypeScript. Essa linguagem tem como propósito adicionar elementos que facilitam o desenvolvimento e a manutenção de um projeto, como sistema nativo de importação de módulos, assinatura de tipo (para métodos) e verificação de tipo em tempo de compilação. Essas características são responsáveis por trazer facilidade e

clareza na importação de módulos e também dão maior previsibilidade ao comportamento do código já que o tipo de informação que trafega em cada método dentro do código se torna explícita.

Todo código em TypeScript, para ser de fato usado, é compilado gerando um código equivalente em JavaScript. Ou seja, o TypeScript é uma ferramenta onde o desenvolvedor consegue escrever um código mais claro e organizado sem abrir mão das funcionalidades nem da compatibilidade com o JavaScript.

4.2.3.3 - Angular

Para facilitar a organização e o desenvolvimento da aplicação foi escolhido o framework Angular. Essa facilidade vem na forma de separação dos elementos da interface em componentes, que são independentes entre si, de modo que cada componente tem um arquivo próprio de modelo (com código HTML), um arquivo de estilo (com código em CSS) e um arquivo de lógica em linguagem TypeScript.

Com isso, é possível trazer ao projeto tanto as vantagens do uso da linguagem TypeScript - código fortemente tipado e clareza na importação de módulos e bibliotecas - quanto a facilidade de organização e geração de código, que são providenciados pelo Angular.

4.3 – Back-end

O *back-end* é responsável por prover informações e serviços necessários para o funcionamento da tela. No caso do escopo deste projeto, têm-se três responsabilidades que serão delegadas ao servidor:

a) Acesso aos dados;
b) Lógica de simulação;
c) Simulação das operações unitárias.

A linguagem escolhida para a camada de servidor foi o Python. A escolha foi feita a partir de três fatores principais:

a) Possui uma Api de acesso ao Matlab. Com isso é possível executar scripts escritos em Matlab dentro de um código de Python (MATHWORKS);
b) Possui alternativas simples para a implementação de serviços, que serão a forma usada para expor as funcionalidades do servidor para qualquer camada de

interface com o usuário. Nesse projeto essa comunicação será implementada usando Flask, um *microframework* usado para expor chamadas de serviço;

c) Linguagem simples e bem difundida no ambiente acadêmico. É a linguagem ensinada nas matérias de computação básica nos cursos de engenharia da UFRJ. (PYTHON, 2017), (GUO, 2014)

4.3.1 – Web Service (Serviço)

Web Service (ou serviço) é um tipo de aplicação utilizada na integração de sistemas e na comunicação entre aplicações diferentes. Essa comunicação é feita a partir de protocolos padronizados, de modo que aplicações contidas em diferentes servidores e desenvolvidas em diferentes linguagens consigam se comunicar usando um protocolo de comunicação Web (como por exemplo o protocolo HTTP).

Um serviço é uma aplicação capaz de expor métodos que serão usados pelos seus consumidores. Uma forma mais simples de entender como o conceito funciona é com um exemplo. A Figura 4.4 apresenta um exemplo de código em Python usando Flask:

```
from flask import Flask
app = Flask(__name__)

@app.route("/hello")
def hello():
    return "Hello LTM!"

app.run()
```

Figura 4.4 - Código Python utilizando *Flask*

Na Figura 4.4, a primeira linha do código refere-se à importação do módulo Flask, responsável por expor o serviço. A quinta linha é referente à implementação do método, no caso, uma simples sequência de texto "Hello World" que será usada para demonstração. Já a quarta linha é referente a rota no servidor correspondente a esse método, ou seja, se o serviço estiver exposto no endereço http://exemplo.com.br, então o método será acessado através do endereço http://exemplo.com.br/hello, seja através de um navegador ou através de uma aplicação. Para ver esse exemplo em prática, pode-se

simplesmente executar o código da Figura 4.4, que irá expor o método "hello", e então acessar a página http://localhost:5000/hello (por padrão o Flask expõe o serviço na porta 5000 do computador). A Figura 4.5 apresenta o retorno do método "hello" sendo acessado segundo o exemplo descrito.

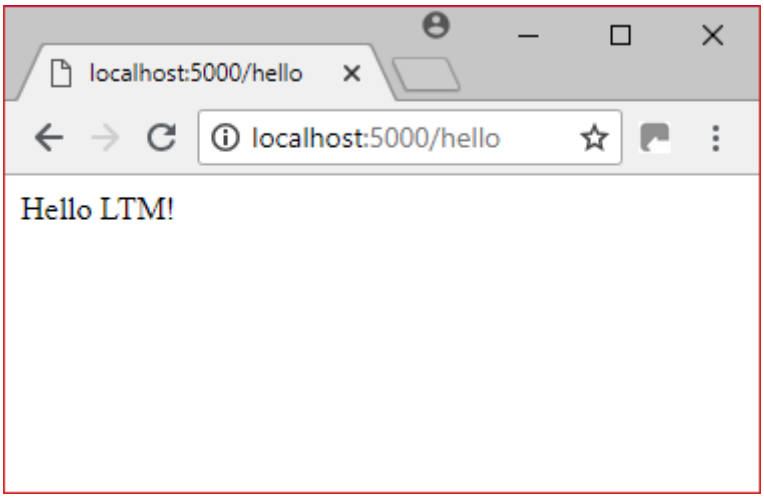

Figura 4.5 – Página de Hello World do serviço em Python usando Flask

A partir desse conceito, pode-se usar o serviço para expor por exemplo o acesso a dados de uma aplicação, sem dar acesso direto ao banco de dados para os consumidores do serviço, e também no caso do LTMSim Open, pode-se executar os scripts escritos em Matlab sem que o usuário da interface possua o Matlab instalado no computador e também sem que o mesmo tenha de fato acesso ao *script* de Matlab. No caso do Flask, ele expõe o serviço usando a arquitetura REST (Representational State Transfer). Além disso, tanto as requisições feitas ao serviço quanto às respostas do mesmo seguem o formato JSON (JavaScript Object Notation).

4.4 – Banco de dados

Para tornar possível a implementação do LTMSim Open, foi necessário o uso de um banco de dados, responsável por armazenar todo tipo de informação que precisa ser persistente - informações de cada tipo de processo e cada um dos modelos que podem ser usados para fazer a simulação, incluindo parâmetros de entrada, saída e conteúdo do código em Matlab. O banco de dados escolhido foi o SQLite, que é um banco de dados

relacional simples e funcional, que não requer nenhum tipo de instalação ou configuração complexa, de modo que uma vez instalado é possível acessá-lo abrindo o arquivo com extensão *.db que armazena as informações.

4.5 – Comportamento - Interface

A interface do LTMSim Open foi planejada para operar com duas telas principais. A primeira é um editor de processos, onde é possível navegar por cada um dos processos armazenados na base, e com isso, editar os modelos relacionados a esse tipo de processo e até adicionar novos modelos que podem ser usados para simular o processo.

4.5.1 - Editor de modelos

O editor fluxograma permite a visualização e edição dos modelos de operações que poderão ser criados pelo usuário ou modificar modelos existentes a partir de um banco de dados. No caso, as operações unitárias são separadas em função de seus tipos principais: britagem, moagem, classificação, separação e fluxos, conforme apresentado na Figura 4.6.

No editor, visando maior alcance da ferramenta LTMSim Open, as operações são definidas por seus termos escritos na língua inglesa: "*Crushing*", "*Grinding*", "*Classification*", "*Separation*" e "*Flow*". Essas classificações são baseadas no conteúdo da base de dados e não tem nenhuma interferência na forma como o simulador irá tratar a lógica de simulação do processo, mas esse tipo de organização serve para facilitar a navegação do usuário pela interface.

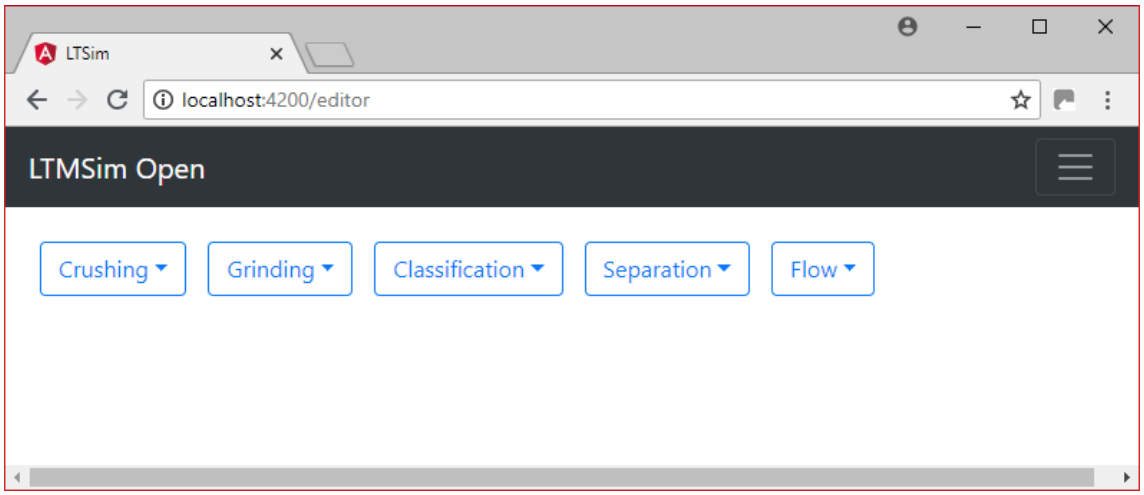

Figura 4.6 – Tela inicial do editor de processos

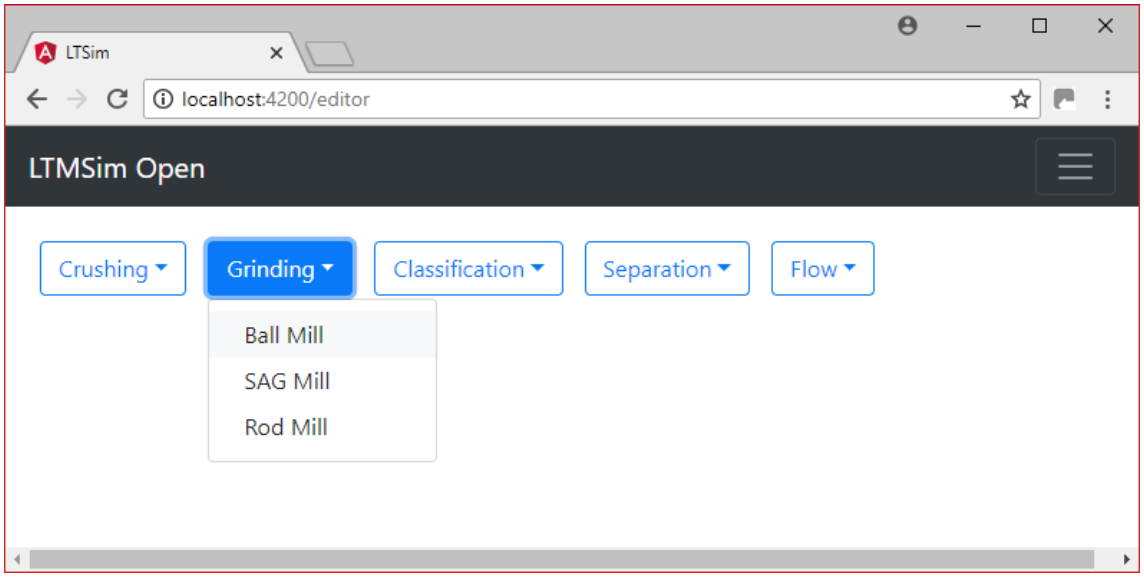

Figura 4.7 – Editor de processos: selecionando um processo

Cada processo que representa uma operação unitária pode ter diversos parâmetros, que podem ser adicionados ou deletados. Um parâmetro tem quatro características principais:

a) "Key" é o nome do parâmetro dentro do código de simulação;
b) "Name" é o nome do parâmetro escrito de forma de fácil entendimento. É esse nome que irá identificar o parâmetro na tela de parametrização do processo;

c) "Unit" se refere ao tipo de unidade a qual o parâmetro é medido. Serve para indicar na interface a unidade do parâmetro a ser inserido;

d) "Type" se refere ao tipo de variável. No momento esse campo não tem impacto na lógica do simulador, mas o campo foi adicionado ao modelo com a intenção de possibilitar que o modelo trate de forma diferente diversos tipos de variáveis (strings, números, enumeradores e distribuições).

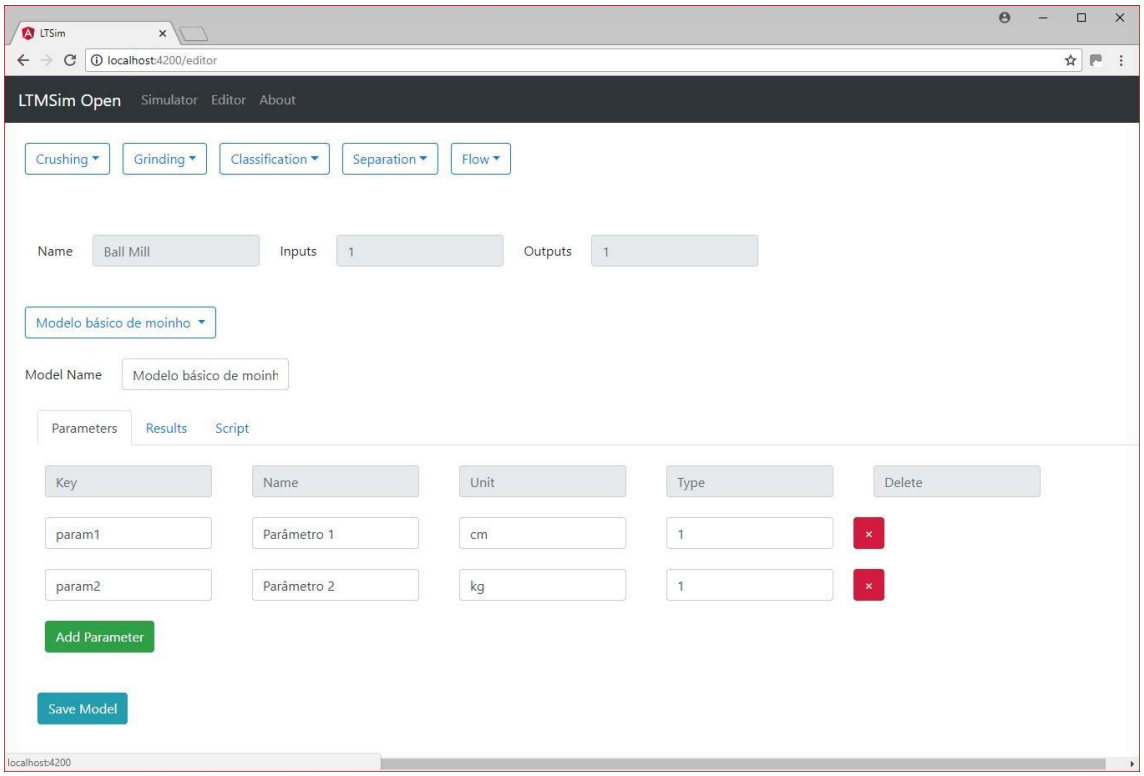

Figura 4.8 – Edição dos parâmetros do processo

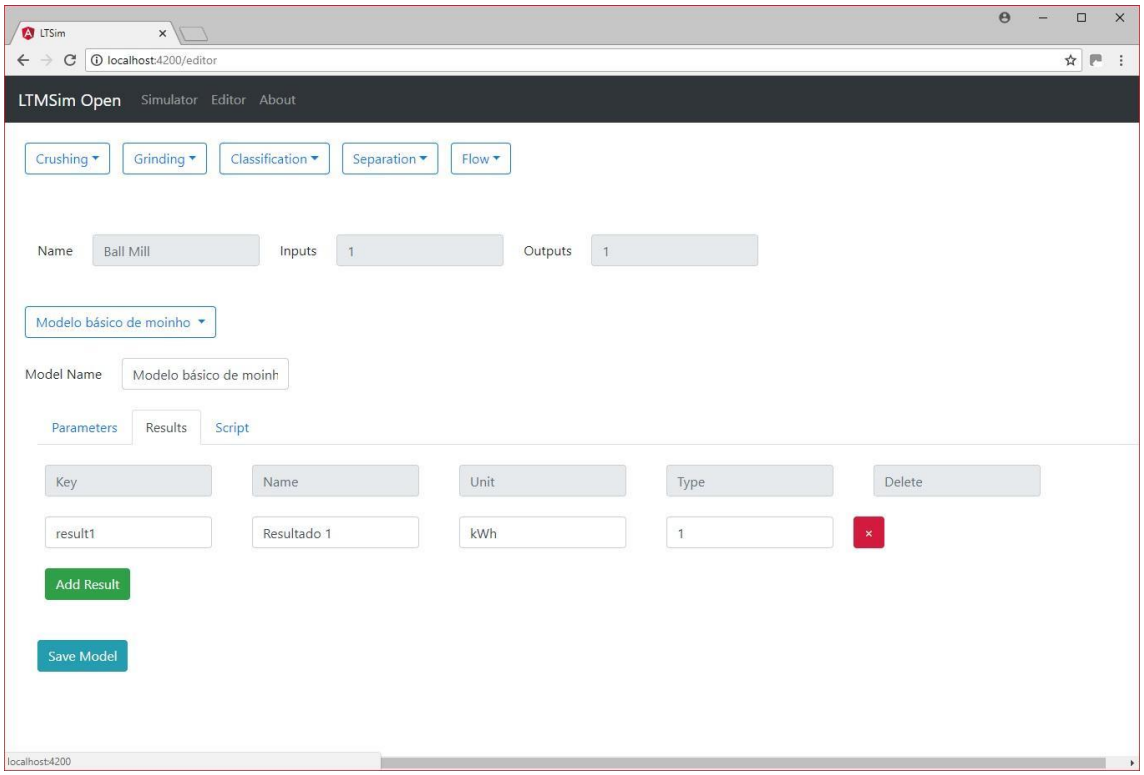

Figura 4.9 – Edição dos resultados que saem do processo

Os resultados também podem ser editados e seguem uma lógica semelhante à dos parâmetros.

A última aba mostrada no exemplo é o editor de código. Neste editor é possível inserir um código em Matlab que será executado ao simular o modelo. Essa aba é dividida em três partes:

a) Declaração de variáveis relevantes. O script recebe os parâmetros de entrada divididos em dois grupos, um dedicado a descrever a corrente de entrada (que todo processo recebe) e outro grupo dedicado a descrever os parâmetros característicos do modelo (definidos na aba anterior);
b) Corpo do código onde toda a lógica do script é programada;
c) Retorno do código, onde as variáveis de saída são definidas. Novamente divididas em dois grupos: os resultados (descritos na aba anterior) e as correntes de saída.

Ao terminar de configurar o modelo, o usuário pode clicar no botão "Save Model" para persistir as configurações feitas na base de dados do servidor.

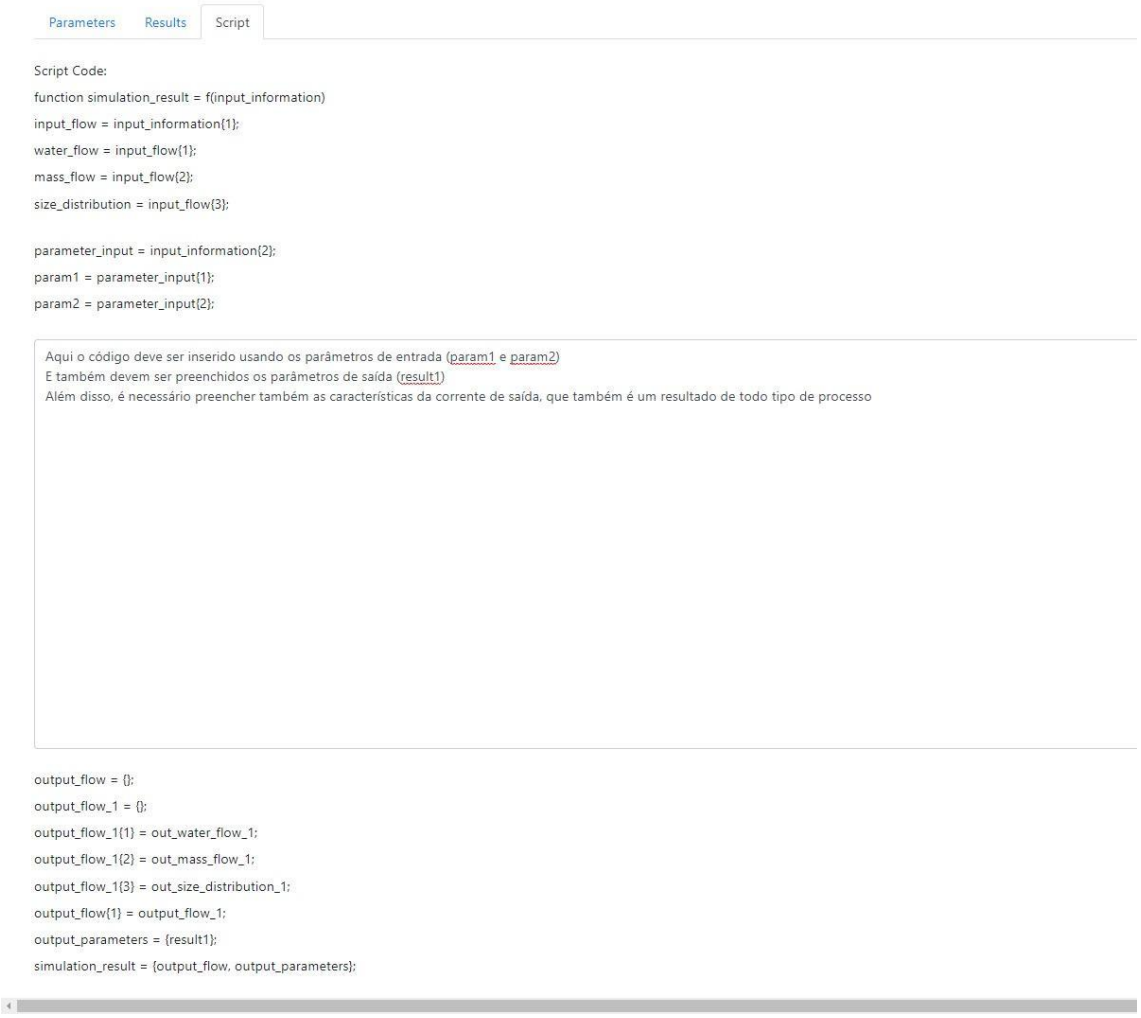

Figura 4.10 – Edição do script de Matlab que executa o processo

4.5.2 - Editor de fluxograma

No editor de fluxograma o usuário pode criar diferentes tipos de fluxograma e executar simulações dos mesmos.

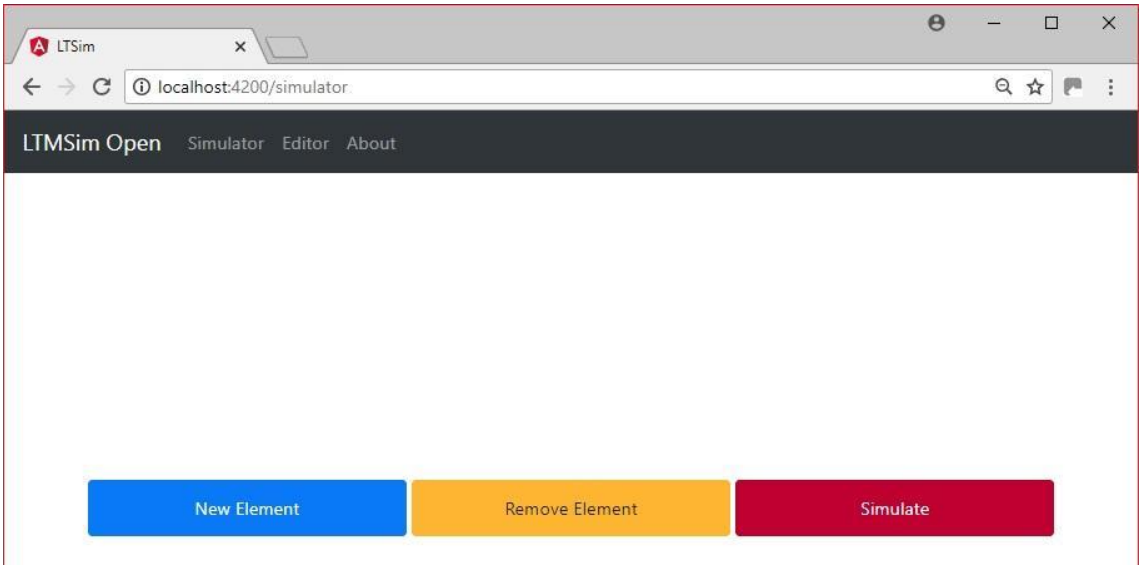

Figura 4.11 – Tela inicial do fluxograma

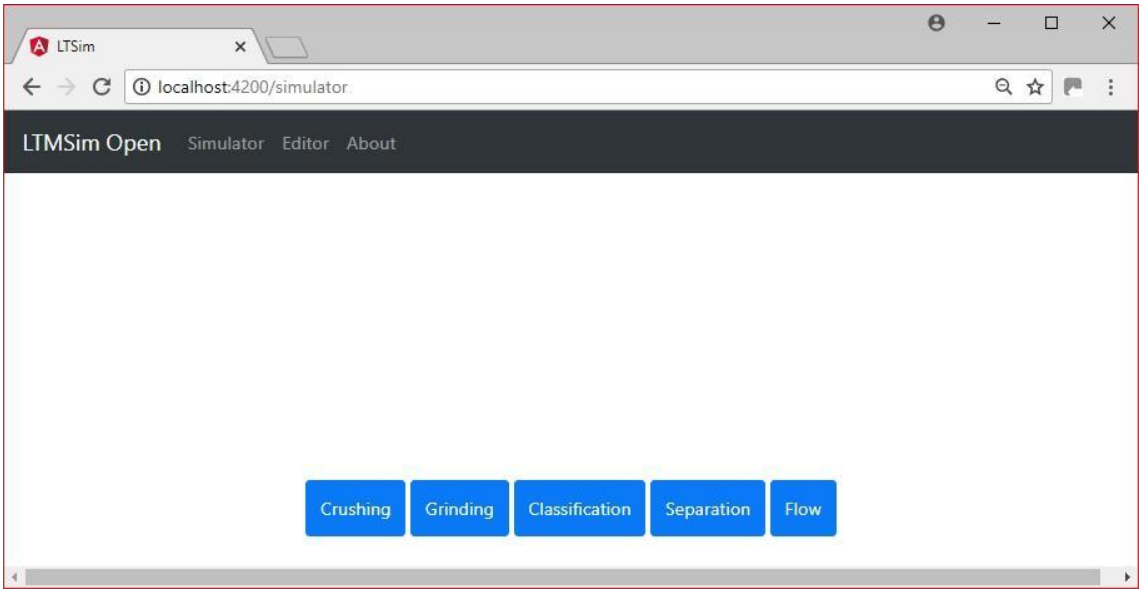

Figura 4.12 – Tela após clicar em New Element

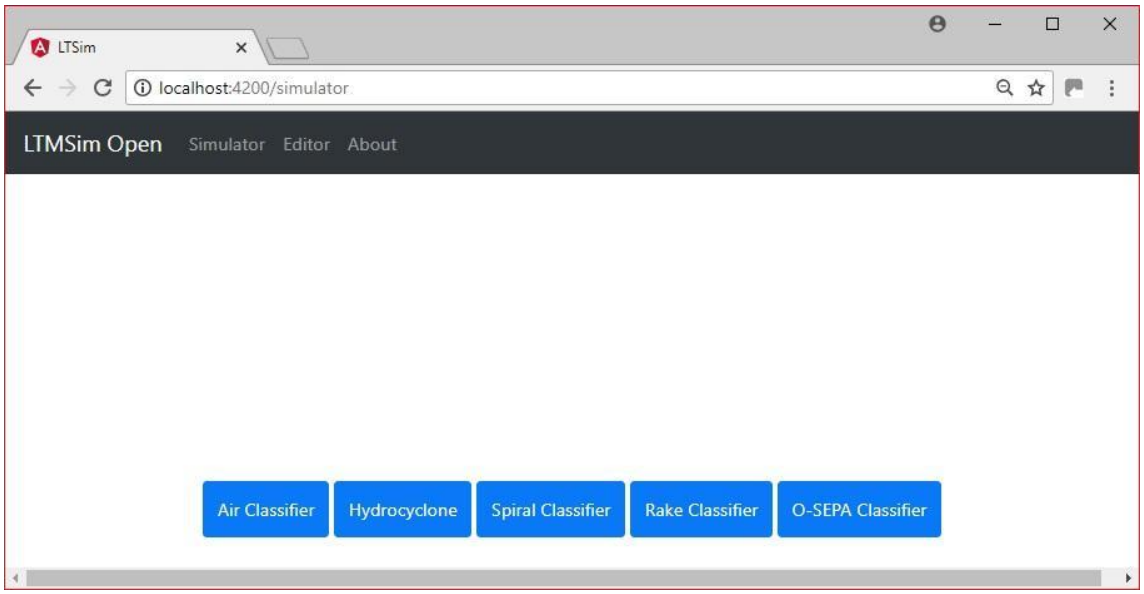

Figura 4.13 – Tela após escolher um tipo de processo

Após adicionar processos unitários ao fluxograma, é possível ligar os processos, conectando cada uma das operações unitárias do fluxograma.

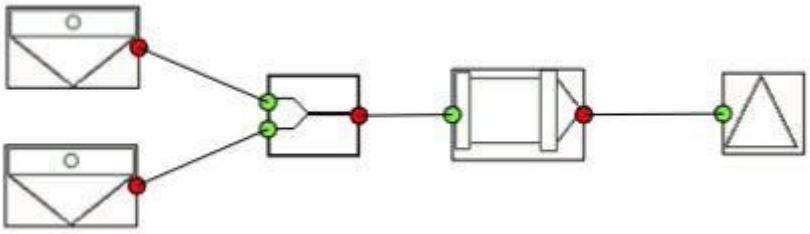

Figura 4.14 – Fluxograma simples (sem nenhum ciclo fechado)

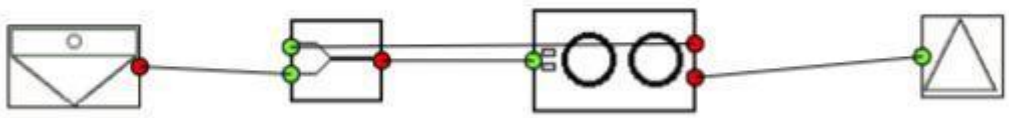

Figura 4.15 –Fluxograma complexo (com um ciclo fechado), onde parte do produto do terceiro processo volta para o segundo processo

Para cada operação unitária na tela do fluxograma, é necessário acessar a tela de detalhes do processo, onde é escolhido o modelo responsável pela simulação do processo e também onde são inseridos os parâmetros que serão usados na simulação.

Após definir os parâmetros de entrada de todas as operações unitárias do processo, pode-se clicar no botão de Simular. O botão de simular acessa o serviço contido no servidor e faz uma requisição pedindo a simulação do fluxograma definido na tela. Ao realizar essa requisição, a interface envia dois vetores como parâmetros para o servidor: um vetor com as correntes do fluxograma e outro vetor com as operações unitárias descritas no mesmo. Com essas informações o servidor é capaz de executar a simulação e retornar um resultado. O resultado de cada operação unitária pode ser visto na tela de detalhes (a mesma onde foram definidos os parâmetros dos processos).

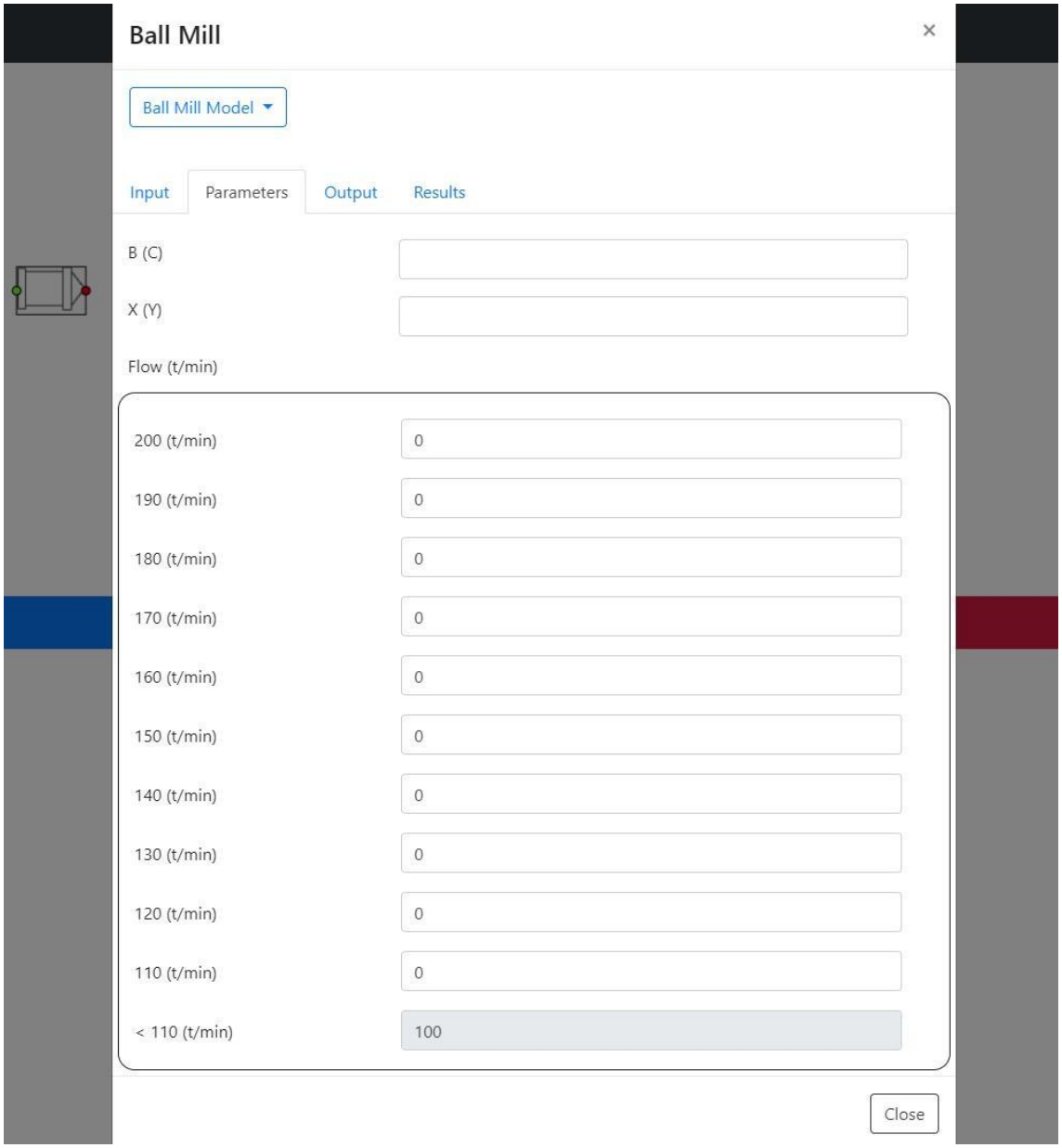

Figura 4.16 - Fluxograma complexo (com um ciclo fechado), onde parte do produto do terceiro processo volta para o segundo processo

4.6 - Comportamento - Serviço

O serviço é dividido em dois caminhos principais: métodos de acesso a dados e métodos de simulação. Dentro desses dois subgrupos o serviço é capaz de realizar quatro requisições diferentes:

a) Acesso a dados:
   - *get process types*: retorna os diferentes tipos de processo guardados na base de dados;
   - *get process by id*: dado um id (número correspondente a um processo), o serviço retorna as informações referentes a esse processo, junto com os modelos relacionados ao mesmo;
   - *post model*: salva as informações referentes a um modelo. Tanto as informações do modelo quanto a identificação de qual modelo salvar vêm da interface ao realizar a requisição.

b) Lógica de simulação:
   - *simulate*: executa a simulação, de acordo com os relacionamentos e processos informados ao chamar o serviço.

As rotinas de acesso a dados são simples operações de criação, leitura e atualização, onde as tabelas referentes a cada tipo de objeto trafegado pela aplicação.

4.6.1 - Simulação

A simulação segue um algoritmo relativamente simples e eficiente. No primeiro momento o serviço recebe um vetor com relacionamentos (contendo os processos os quais o relacionamento conecta e a corrente que é trafegada pelo relacionamento) e os processos (cada operação unitária do fluxograma).

Para possibilitar a simulação, é necessário que o algoritmo trate os dados de entrada, de modo que eles possam passar pela simulação de forma coerente.

O primeiro tratamento é feito no vsetor e correntes, onde tem-se informações referentes aos processos conectados pela corrente e o fluxo que passa pela mesma (com isso o serviço entende também quais correntes são conhecidas). Então essas correntes são passadas para o Ciclor, que decide quais correntes devem ser alimentadas (com

fluxos nulos de água e sólidos) para que a simulação se torne possível. A partir dessa decisão, as correntes necessárias são alimentadas.

O próximo tratamento é feito no vetor de processos, que deve ser organizado de modo que cada processo seja simulado somente quando as correntes de entrada do mesmo forem conhecidas. Com isso, os processos são organizados da seguinte forma:

a) Variáveis iniciais;
    - Lista P inicial de processos, contendo todos os processos de forma não ordenada;
    - Lista C que contém apenas as correntes conhecidas.
b) Cria-se uma lista L que irá conter os processos ordenados. Essa lista começa vazia;
c) Iteração pela lista P. Para cada processo em P, verifica se as correntes de entrada do processo são conhecidas. Nesse caso, o processo é adicionado à lista L e as correntes de saída do processo são adicionadas à lista C;
d) Após diversas iterações da rotina (c), eventualmente a lista L terá todos os processos do fluxograma e a lista C terá todas as correntes.

Com os processos devidamente organizados e as correntes devidamente alimentadas, é criada uma lista que irá conter todas as rodadas de simulação dos processos. A cada rodada de simulação uma nova entrada será adicionada nessa lista, contendo os estados dos processos e das correntes ao final de cada ciclo de simulação.

Além disso, ao final de cada rodada, as duas últimas rodadas são comparadas para verificar a convergência da simulação. Essa comparação é feita calculando uma variável de erro entre as duas simulações. Se essa variável de erro estiver dentro de um limiar aceitável, então a simulação é dada como completa e o resultado é retornado para a interface de simulação.

O cálculo do erro é feito comparando cada variável de cada processo e de cada corrente. O erro calculado é a maior diferença entre duas variáveis de um mesmo processo entre as duas rodadas. Por exemplo, se estivermos simulando um fluxo com dois processos, P_1 e P_2. P_1tem como parâmetro a variável A, enquanto P_2tem como parâmetros as variáveis X e Y. O erro calculado entre os ciclos 0 e 1 é igual ao valor máximo dentre as seguintes diferenças: $\frac{(A_1 - A_0)^2}{{A_0}^2}$, $\frac{(X_1 - X_0)^2}{{X_0}^2}$, $\frac{(Y_1 - Y_0)^2}{{Y_0}^2}$. Foi considerado

calcular o erro a partir da soma os da média dos erros de cada parâmetro, entretanto essa forma foi escolhida por três motivos principais:

a) O método de fazer um somatório de todas as diferenças de todos os parâmetros sofre com o viés de que quanto maior for o número de processos, maior a quantidade de termos no somatório, fazendo com que o critério de convergência fique mais difícil de ser alcançado em processos complexos;
b) O efeito contrário seria alcançado caso a alternativa escolhida fosse a média das diferenças calculadas, de modo que se um parâmetro ainda estiver longe da convergência, mas com todos os outros parâmetros já tendo alcançado a convergência, então em fluxos grandes é possível que a convergência seja alcançada mesmo tendo um dos parâmetros ainda muito acima do limiar de erro;
c) Com essas situações em mente, o critério de comparar o maior erro percentual com um dado limite de convergência se mostra mais coerente, porque garante que cada variável calculada está dentro de um limite aceitável de variação entre um ciclo e outro.

Após a simulação alcançar o critério de convergência, a mesma é finalizada e o resultado é retornado para a tela da interface.

4.7 – Estudos de casos

4.7.1 - Simulação simples de HPGR

Um cenário considerado para demonstração do LTMSim Open foi a implementação do modelo de HPGR recentemente proposto por CAMPOS (2018). Para realizar esse teste, foi construído um fluxograma simples com um HPGR recebendo uma alimentação e gerando dois produtos - um deles correspondente a parcela que passou pelas bordas dos rolos do HPGR e outro que passou pelo meio dos rolos do HPGR - que são misturados em um misturador, formando um produto final como apresentado na Figura 4.17, que apresenta o fluxograma construído na interface de edição de fluxogramas do LTMSim Open. Após a realização da simulação, os resultados foram comparados com aqueles obtidos no trabalho original de CAMPOS (2018).

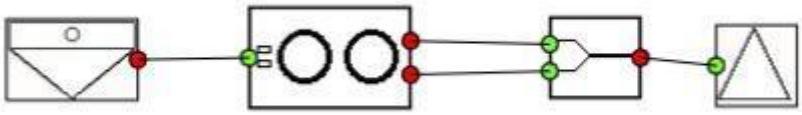

Figura 4.17 - Fluxograma simples com HPGR

4.7.2 - Simulação de circuito com HPGR e reciclo do produto das bordas

Outro cenário considerado para demonstração do LTMSim Open foi o de um fluxograma complexo, com ciclos fechados e retroalimentação. Nesse caso o HPGR não é alimentado diretamente pelo fluxo de alimentação do sistema. Dessa vez o HPGR é alimentado por um misturador, que por sua vez é alimentado tanto pelo fluxo de alimentação do sistema quanto pelo próprio HPGR, ou seja, parte do produto que sai do HPGR (a parte do produto que foi submetida às bordas dos rolos) é usada novamente na alimentação do equipamento, de acordo com o fluxograma apresentado na figura 4.18.

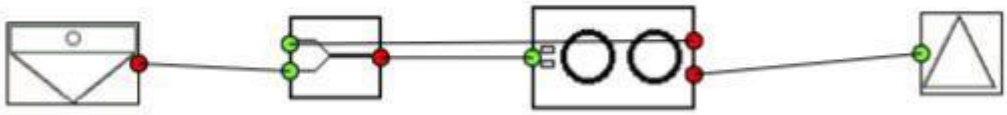

Figura 4.18 - Fluxograma complexo com HPGR, onde o HPGR é alimentado com o próprio produto

4.7.3 - Parametrização

Ambos os casos de teste foram testados usando os mesmos parâmetros, que são:

Tabela 4.1 - Corrente de alimentação - Vazão de sólidos e de água

Parâmetro	Valor
Vazão de água	0 t/h
Vazão de sólidos	100 t/h

Tabela 4.2 - Corrente de alimentação - Distribuição granulométrica da corrente

Abertura de peneira (mm)	Retido (%)	Passante acumulado (%)
2	0,04	99,96
1	0,03	99,93
0,5	0,03	99,90
0,3	0,94	98,96
0,15	7,51	91,45
0,106	11,49	79,96
0,075	17,29	62,67
0,063	9,98	52,69
0,045	18,97	33,72
0,038	8,37	25,35
0,02	19,62	5,73
0,01	5,05	0,68
0,005	0,66	0,02
0,002	0,02	0,00
0,001	0,00	0,00
< 0,001	0,00	0,00

Tabela 4.3 - Parâmetros HPGR - Modelo de capacidade

Parâmetro	Valor
a	40,817
b	-0,06733

Tabela 4.4 - Parâmetros HPGR - Fator de ajuste para o ângulo de captura

Parâmetro	Valor
C	2,6

Tabela 4.5 - Parâmetros HPGR - Condições operacionais

Parâmetro	**Valor**
Diâmetro do rolo	1 m
Comprimento do rolo	0,32 m
Pressão de operação	54 bar
Massa específica	5,07 t/m³
Densidade aparente	3 t/m³
Abertura de trabalho	0,011 m
Velocidade Linear dos rolos	0,3 m/s
Velocidade da banda	1 m/s
Penetração dos studs no flake	0,00173 m
Fração da área superficial do rolo onde há camada autógena	0,65

Tabela 4.6 - Parâmetros HPGR - Função de quebra

Parâmetro	**Valor**
n_1	0,800518
n_2	2,386170
K	0,736624
y_0	0,10078 mm
n_3	0,27

Tabela 4.7 - Parâmetros HPGR - Função de seleção

Parâmetro	**Valor**
$\boldsymbol{\xi}_1$	-1,843836
$\boldsymbol{\xi}_2$	-0,351003
s(1)	0,172386 t/kWh

Capítulo 5

Resultados

5.1 - Cadastro dos modelos

5.1.1 - Alimentação e pilha

Tanto a alimentação quanto a pilha seguem um modelo muito simples: não tem nenhum parâmetro de entrada ou saída referente ao modelo, recebe apenas a corrente de entrada, com vazão de água, vazão de sólido e distribuição granulométrica dos sólidos.

A implementação do modelo também é simples, já que a corrente de saída dos processos é exatamente igual à corrente de entrada.

O vetor que compõe a corrente é composto pelos seguintes elementos:

a) Fluxo de água (em t/h)
b) Fluxo de sólidos (em t/h)
c) Vetor de distribuição granulométrica, onde que cada linha é composta por:
    - Abertura da peneira;
    - Porcentagem retida.

5.1.2 - HPGR

O HPGR tem um modelo mais complexo, com diversos parâmetros de entrada e variáveis de saída. Com isso, foi cadastrado no simulador o modelo proposto por CAMPOS (2018) recebendo os parâmetros descritos na seção 4.7.3.

5.2 - Estudo de caso

5.2.1 - Simulação simples de HPGR

Foram considerados os parâmetros utilizados no modelo de Campos (2018), listados na seção 4.7. Após executar a simulação, foram obtidos os seguintes resultados, de acordo com as Tabelas 5.1, 5.2:

Tabela 5.1 - Correntes de saída - Vazão de sólidos e de água

	Total	Borda	Meio
Vazão de água (t/h)	0	0	0
Vazão de sólidos (t/h)	100	50	50

Tabela 5.2 - Correntes de saída - Distribuição granulométrica das correntes de saída

	Retido (%)			**Passante acumulado (%)**		
Abertura de peneira (mm)	Total	Borda	Meio	Total	Borda	Meio
2	0,04	0,04	0,04	99,96	99,96	99,96
1	0,03	0,03	0,02	99,94	99,93	99,94
0,5	0,02	0,03	0,02	99,91	99,91	99,92
0,3	0,54	0,66	0,42	99,37	99,24	99,49
0,15	4,03	5,06	3,01	95,33	94,19	96,48
0,106	6,91	8,34	5,48	88,43	85,85	91,01
0,075	11,88	13,70	10,05	76,55	72,14	80,95
0,063	8,26	8,96	7,56	68,29	63,19	73,39
0,045	16,97	17,86	16,08	51,32	45,32	57,31
0,038	9,38	9,17	9,58	41,94	36,15	47,73
0,02	24,97	23,30	26,63	16,98	12,85	21,10
0,01	11,45	9,13	13,78	5,52	3,72	7,32
0,005	3,39	2,37	4,41	2,14	1,36	2,91
0,002	1,37	0,87	1,87	0,76	0,48	1,04
0,001	0,40	0,26	0,55	0,36	0,23	0,49
< 0,001	0,36	0,23	0,49	0,00	0,00	0,00

A partir dos resultados de porcentagem retida por tamanho de partícula apresentados na tabela 5.2 foi feita uma interpolação entre os valores de passante acumulado e as aberturas de peneira. A partir dessa interpolação foram obtidos os valores de P_{80} do total, da borda e do meio como sendo iguais a 0,084 mm, 0,092 mm e 0,073 mm, respectivamente, ilustrando que o produto do meio dos rolos é o mais fino de todos.

A Figura 5.1 mostra as distribuições granulométricas da alimentação e de saída da simulação:

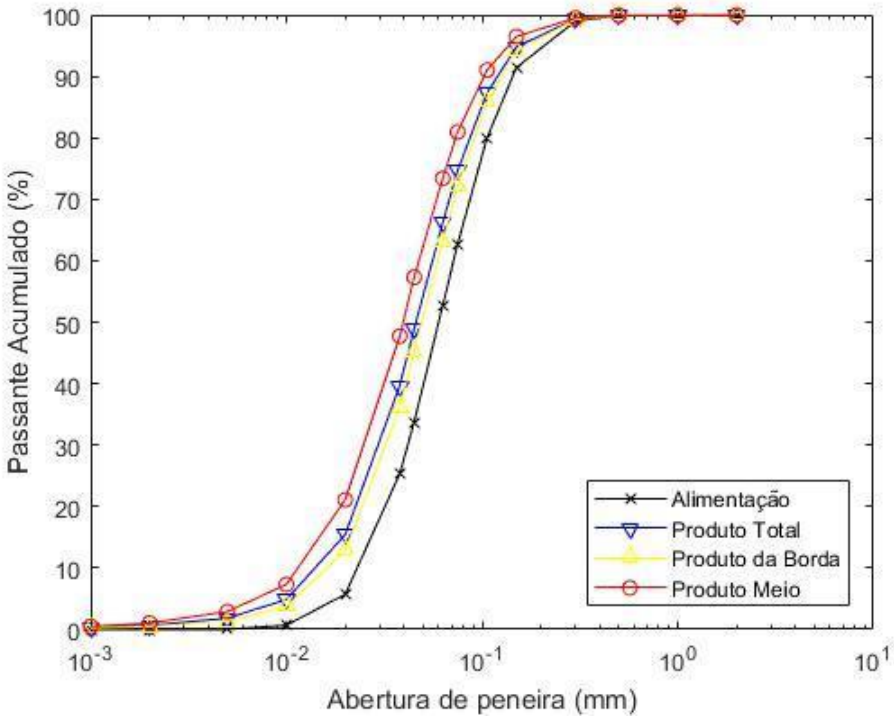

Figura 5.1 - Passante acumulado da alimentação e dos produtos obtidos através da simulação

Como esperado, os produtos obtidos apresentam uma granulometria mais fina que a alimentação. Além disso, o produto que passou pela região central dos rolos apresenta a granulometria mais fina dentre todos os produtos, enquanto o produto da borda apresentou a mais grossa.

A fim de verificar se o modelo de Campos (2018) foi implementado corretamente no LTMSim Open, compara-se o resultado com aquele obtido ao utilizar o código original de CAMPOS (2018), fora da plataforma LTMSim Open.

As figuras 5.2, 5.3 e 5.4 mostram a comparação entre o passante acumulado de cada produto obtido na simulação feita no LTMSim e aquele obtido por CAMPOS (2018). A Figura 5.2 mostra a comparação entre o passante acumulado entre o produto total obtido. Já Figura 5.3 mostra a comparação entre o passante acumulado entre o produto das bordas dos rolos. A Figura 5.4 mostra a comparação entre o passante acumulado entre o produto do meio dos rolos:

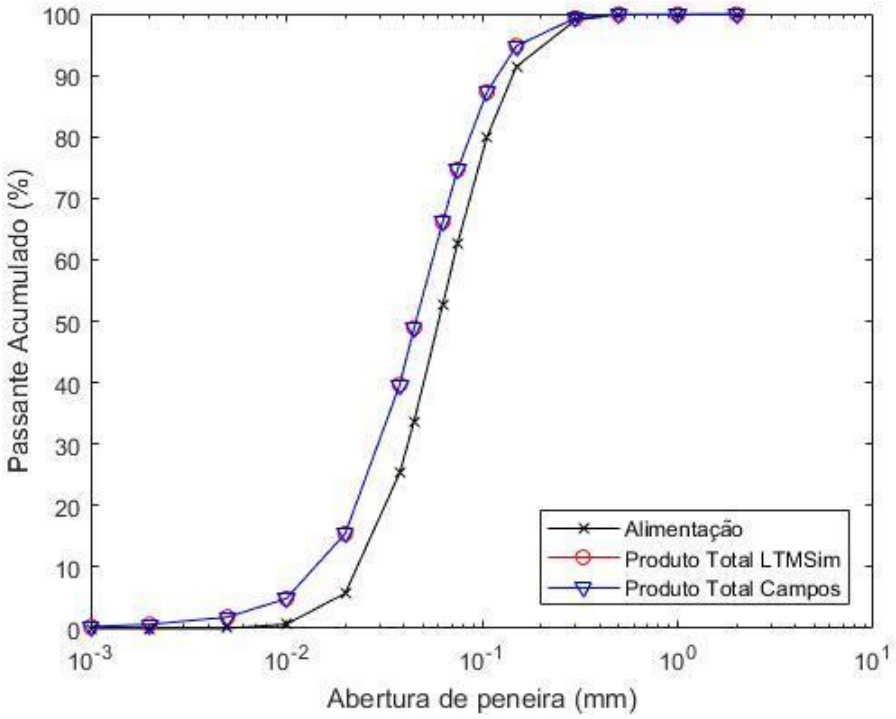

Figura 5.2 - Comparação entre os passantes acumulados da alimentação e do produto total obtido pelas duas simulações

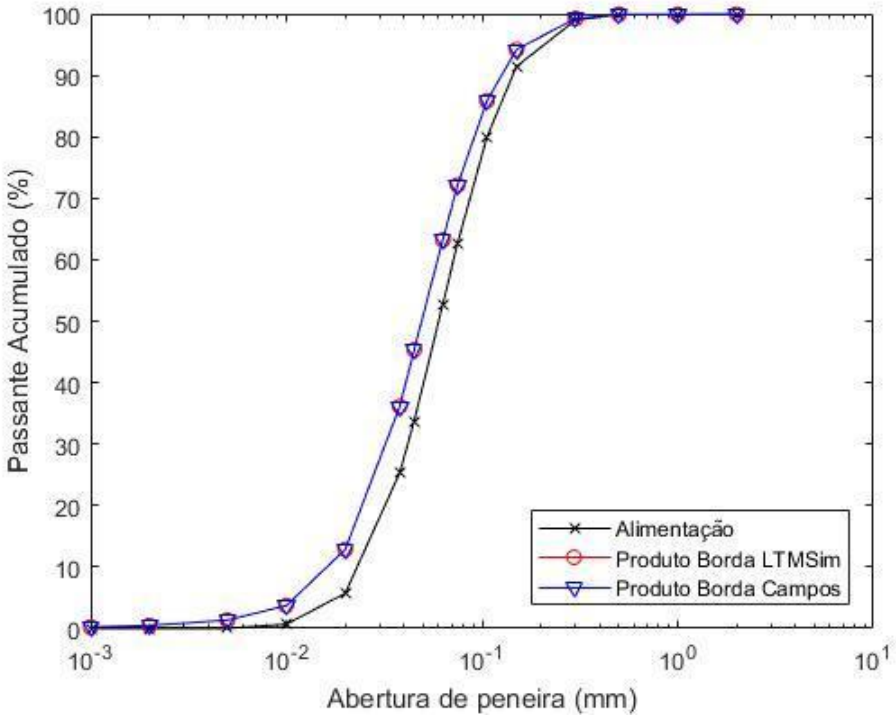

Figura 5.3 - Comparação entre os passantes acumulados da alimentação e do produto das bordas dos rolos obtido pelas duas simulações

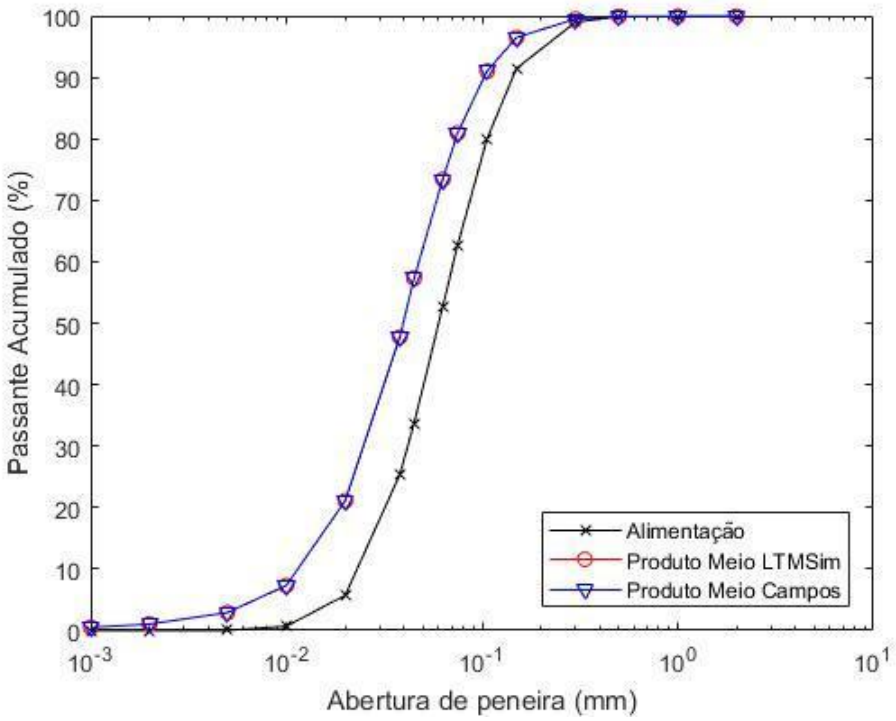

Figura 5.4 - Comparação entre os passantes acumulados da alimentação e do produto do meio dos rolos obtido pelas duas simulações

Como esperado, as curvas de todos os gráficos de comparação se sobrepõem. Isso ocorre porque ambas as simulações foram feitas usando a mesma implementação do mesmo modelo e com os mesmos parâmetros, mostrando que o modelo proposto foi incorporado com sucesso no sistema.

A Tabela 5.3 apresenta a comparação dos valores calculados para área superficial do produto do HPGR pela simulação do LTSim com os valores obtidos por CAMPOS (2018) na mesma simulação:

Tabela 5.3 - Resultados do HPGR. Comparação entre LTMSim e CAMPOS (2018)

Parâmetro	**LTMSim**	**CAMPOS (2018)**
Área Superficial (Total) (cm²/g)	772,97	772,97
Área Superficial (Bordas) (cm²/g)	708,96	708,96
Área Superficial (Meio) (cm²/g)	915,16	915,16

Novamente como esperado, os resultados obtidos tanto pelo LTMSim Open quanto pelo modelo de CAMPOS (2018) foram os mesmos, por implementarem o mesmo modelo. Pode-se observar também que a área superficial obtida no meio dos rolos é maior que nas bordas, mostrando que a moagem no meio dos rolos é mais efetiva.

A Figura 5.5 mostra como esses resultados são mostrados na interface do simulador:

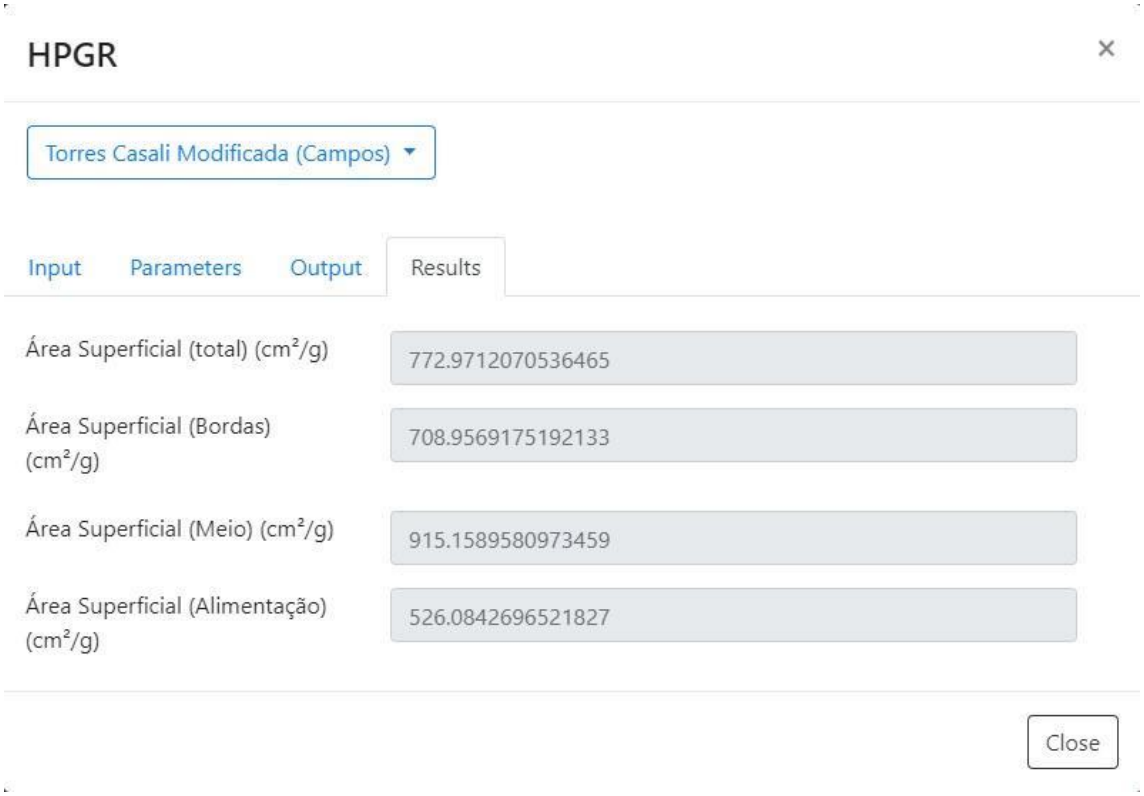

Figura 5.5 - Tela de resultados da simulação do HPGR

5.2.2 - Simulação de circuito com HPGR e reciclo do produto das bordas

Considerados os parâmetros utilizados no modelo de Campos (2018), listados na seção 4.7. Após executar a simulação, foram obtidos os seguintes resultados, de acordo com as Tabelas 5.4 e 5.5:

Tabela 5.4 - Corrente de saída - Vazão de sólidos e de água (ciclo fechado)

	Total	Borda	Meio
Vazão de água (t/h)	0	0	0
Vazão de sólidos (t/h)	100	100	100

Tabela 5.5 - Correntes de saída - Distribuição granulométrica da corrente de saída (ciclo fechado)

	Retido (%)			**Passante acumulado (%)**		
Abertura de peneira (mm)	Total	Borda	Meio	Total	Borda	Meio
2	0,04	0,04	0,04	99,96	99,96	99,96
1	0,02	0,03	0,02	99,94	99,94	99,94
0,5	0,02	0,02	0,02	99,92	99,92	99,92
0,3	0,33	0,51	0,33	99,59	99,40	99,59
0,15	2,27	3,81	2,27	97,32	95,59	97,32
0,106	4,26	6,52	4,26	93,06	89,07	93,06
0,075	8,16	11,21	8,16	84,90	77,86	84,90
0,063	6,49	7,84	6,49	78,41	70,02	78,41
0,045	14,35	16,26	14,35	64,06	53,76	64,06
0,038	9,24	9,18	9,24	54,82	44,58	54,82
0,02	27,60	25,21	27,60	27,22	19,37	27,22
0,01	17,02	12,70	17,02	10,20	6,67	10,20
0,005	6,08	4,06	6,08	4,11	2,61	4,11
0,002	2,65	1,68	2,65	1,46	0,93	1,46
0,001	0,77	0,49	0,77	0,69	0,44	0,69
< 0,001	0,69	0,44	0,69	0,00	0,00	0,00

A partir dos resultados de porcentagem retida por tamanho de partícula apresentados na tabela 5.5 foi feita uma interpolação entre os valores de passante acumulado e as aberturas de peneira. A partir dessa interpolação foram obtidos os valores de P_{80} do total, da borda e do meio como sendo iguais a 0,066 mm, 0,081 mm e 0,066 mm, respectivamente, ilustrando que o produto do meio dos rolos é o mais fino que o da borda e igual ao que chega na pilha final, segundo o fluxograma do processo, somente a região processada no meio dos rolos vai para a pilha, pois o produto das bordas dos rolos sempre é reprocessado

A Figura 5.6 mostra o gráfico com as distribuições granulométricas da alimentação e de saída da simulação. Pode-se observar que agora o produto é exatamente igual ao produto da região do meio dos rolos. Isso ocorre porque todo o

produto da região das bordas é reprocessado, de modo que só o produto da região central dos rolos vai para a pilha final:

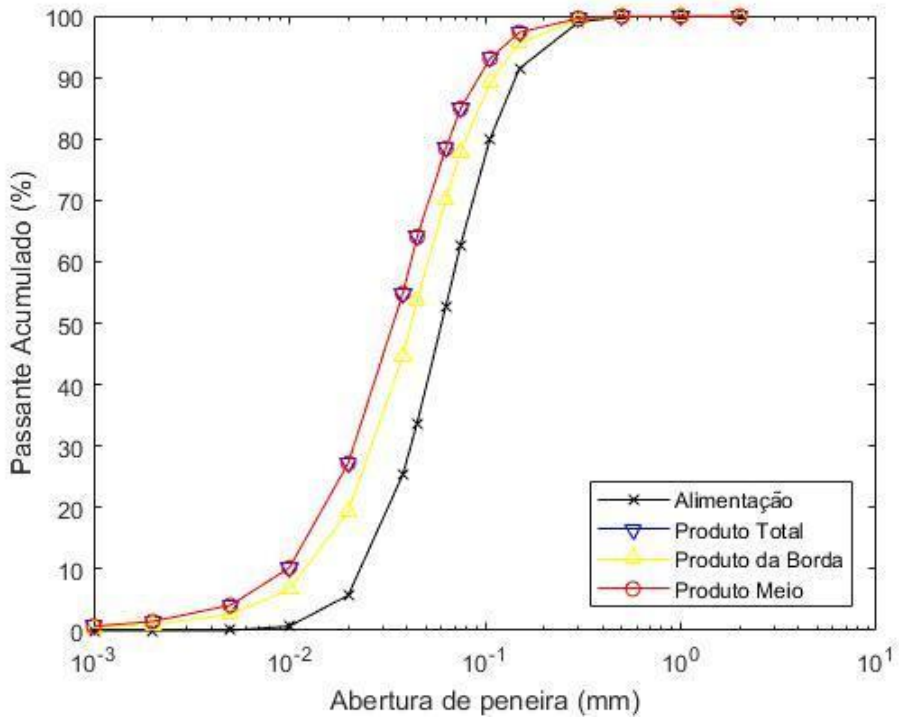

Figura 5.6 - Passante acumulado da alimentação e dos produtos obtidos através da simulação (Ciclo fechado)

As Figuras 5.7, 5.8 e 5.9 mostram a comparação entre o passante acumulado de cada produto obtido no caso de ciclo fechado (do presente estudo de caso) e o produto do caso de ciclo aberto (estudo de caso anterior). A Figura 5.7 mostra a comparação entre o passante acumulado entre o produto total obtido. Já Figura 5.8 mostra a comparação entre o passante acumulado entre o produto das bordas dos rolos. A Figura 5.9 mostra a comparação entre o passante acumulado entre o produto do meio dos rolos:

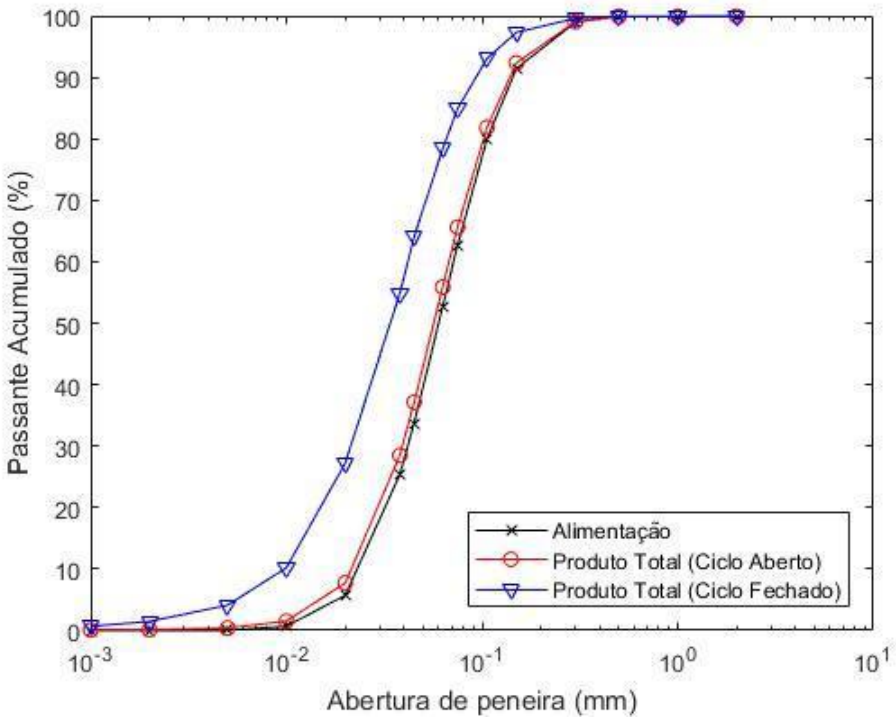

Figura 5.7 - Comparação entre os passantes acumulados da alimentação e do produto total obtido pela simulação do modelo aberto e do modelo fechado

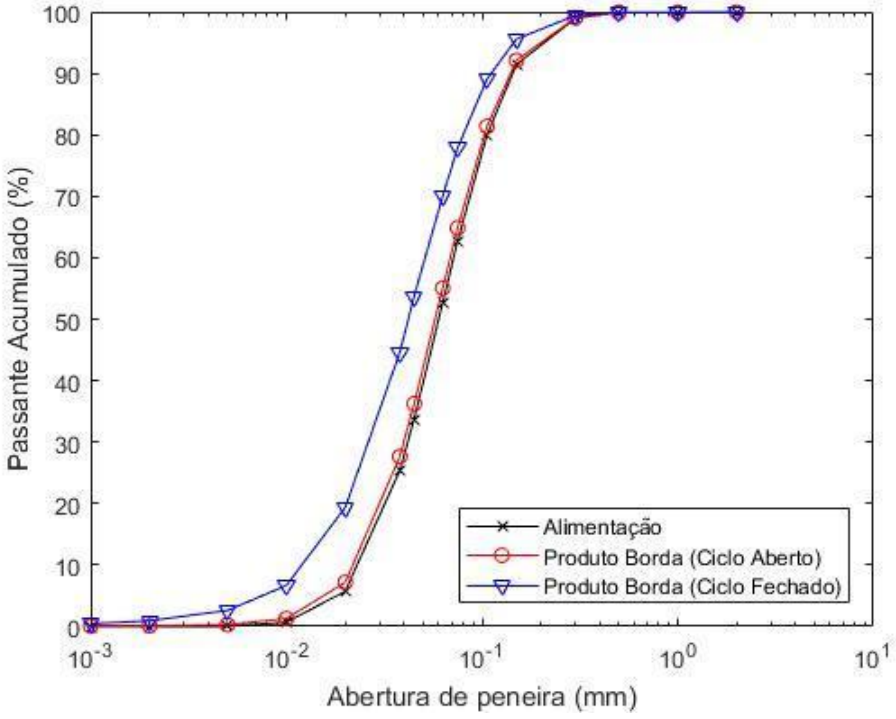

Figura 5.8 - Comparação entre os passantes acumulados da alimentação e do produto das bordas obtido pela simulação do modelo aberto e do modelo fechado

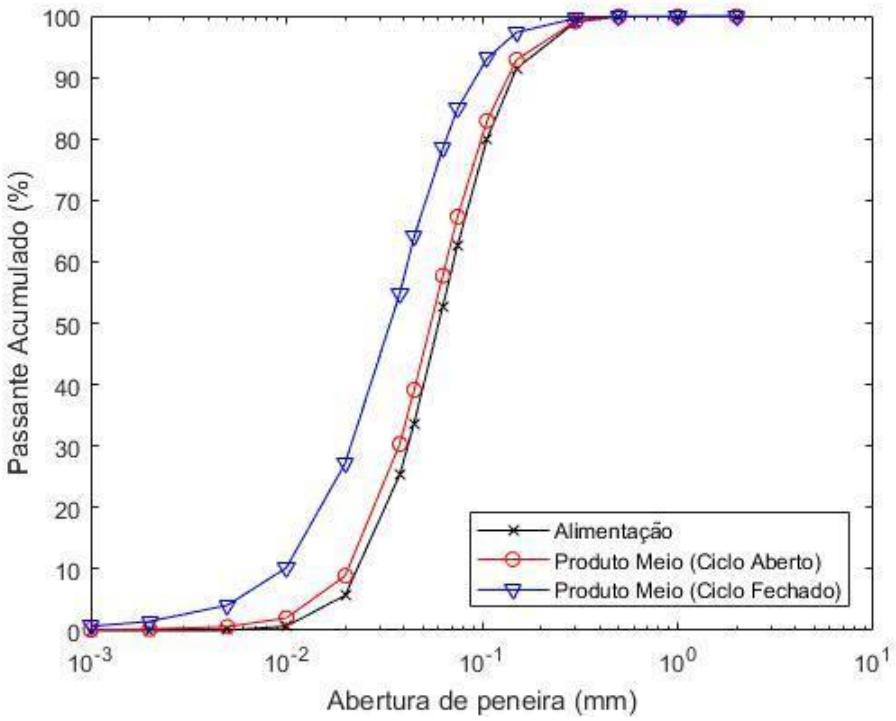

Figura 5.9 - Comparação entre os passantes acumulados da alimentação e do produto do meio obtido pela simulação do modelo aberto e do modelo fechado

Os resultados das comparações estão de acordo com o esperado, já que o processo de ciclo fechado recicla a parte mais grossa do produto do HPGR, que é reprocessado, obtendo uma granulometria mais fina.

A tabela 5.6 apresenta a comparação entre os valores calculados para área superficial do produto do HPGR pela simulação do sistema de ciclo aberto com o sistema de ciclo fechado:

Tabela 5.6 - Resultados do HPGR. Comparação entre sistema de ciclo aberto e ciclo fechado

Parâmetro	**Ciclo Fechado**	**Ciclo Aberto**
Área Superficial (Total) (cm^2/g)	929,58	772,97
Área Superficial (Bordas) (cm^2/g)	869,49	708,96
Área Superficial (Meio) (cm^2/g)	1063,06	915,16

Novamente temos um resultado dentro do esperado, já que o ciclo fechado foi capaz de alcançar uma área superficial ainda maior que a do ciclo aberto, devido ao reprocessamento da parcela mais grossa do produto. Também é importante observar que esses resultados são referentes a operação unitária na qual eles foram calculados. Assim, a área superficial total calculada é referente a mistura entre o produto do meio dos rolos e o produto das bordas dos rolos, sendo diferente do produto que chega na pilha, que é igual ao produto do meio dos rolos somente.

A Figura 5.10 mostra como esses resultados são mostrados na interface do simulador:

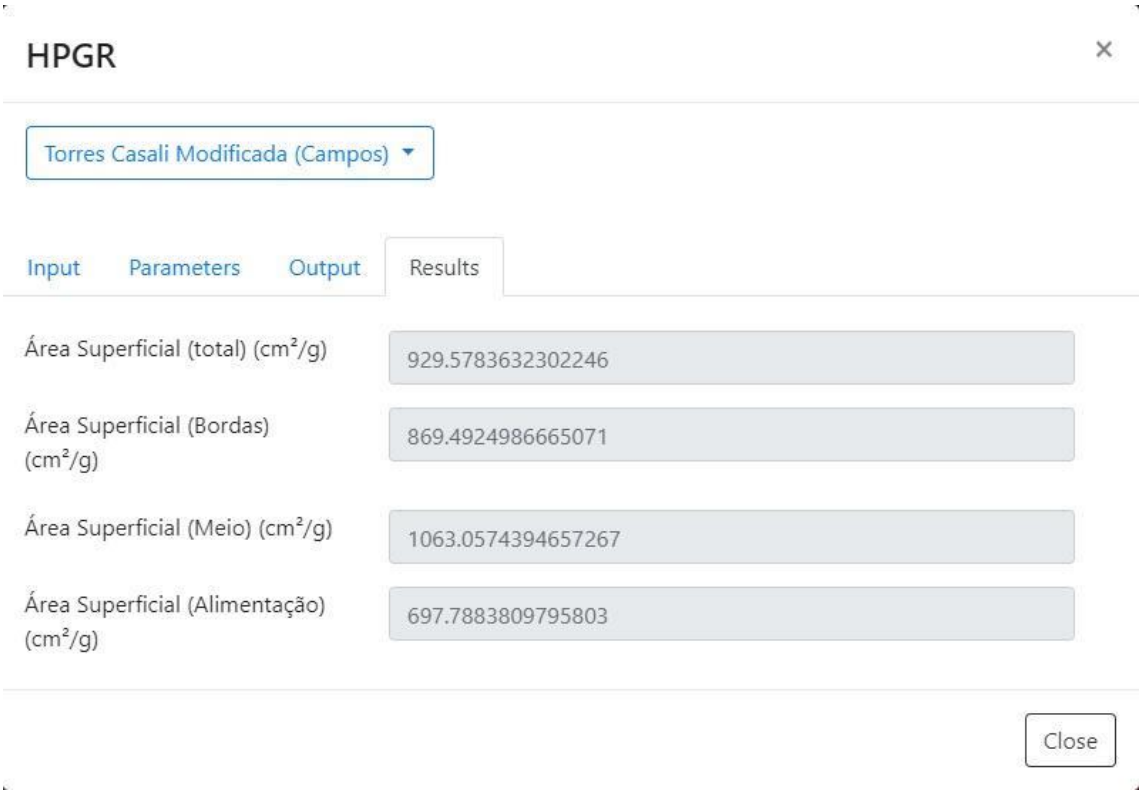

Figura 5.10 - Tela de resultados da simulação do HPGR (Ciclo fechado)

Capítulo 6

Conclusões

Os resultados dos testes mostraram que o simulador foi implementado com sucesso e que o mesmo é capaz de implementar modelos e realizar simulações de acordo com a necessidade do usuário.

Neste trabalho foi desenvolvido um simulador que permite cadastrar diversos modelos para simular cada tipo de processo, de modo que cada modelo tem seus próprios parâmetros, tanto de entrada quanto de saída. Esses parâmetros serão usados posteriormente pelo código do modelo, que também é cadastrado dentro da aplicação, que consiste em um código em Matlab que será executado quando a simulação for requisitada.

Com os modelos devidamente cadastrados, o programa é capaz de montar os fluxogramas definidos pelo usuário e com isso, receber os parâmetros de cada modelo (previamente designados no momento do cadastro). O simulador executa a simulação do fluxograma de processo, retornando os resultados da simulação.

Para demonstrar o comportamento do simulador, foram feitos dois testes envolvendo o modelo de HPGR proposto por CAMPOS (2018). Os resultados do primeiro teste foram comparados com os obtidos no trabalho do autor do modelo, de modo que o simulador conseguiu chegar aos mesmos resultados que CAMPOS (2018).

Ao simular o segundo caso de teste, os resultados foram comparados com os resultados obtidos no primeiro caso. Foi observado que no segundo caso os produtos apresentam granulometria mais fina que no primeiro, o que era o resultado esperado devido ao reprocessamento do produto das bordas do equipamento.

Tendo o simulador implementado, o mesmo pode ser usado para auxiliar futuros trabalhos, possibilitando integrar modelos recém desenvolvidos ou ainda em desenvolvimento a fluxogramas de processos e entender como esses modelos se comportam em diferentes situações.

Tendo em vista que o desenvolvimento de um simulador de processos é um trabalho com escopo extenso, de forma que ainda existem diversas funcionalidades que

ainda podem ser adicionadas. Funcionalidades como calibração dos modelos, balanço de massa e maior flexibilidade na definição dos tamanhos de peneira usados na parametrização dos modelos ainda não estão implementados. Além disso, ainda há necessidade de melhorias na interface do simulador, de forma a aprimorar a usabilidade da aplicação e facilitar a visualização dos dados.

Bibliografia

ALVES, S. H., *Implementação de Algoritmo de Resolução Sequencial no Simulador de Processos LTMSim*. Projeto final de curso. Universidade Federal do Rio de Janeiro, 2011

ARAÚJO, R. de P., *SEPARAÇÃO DE MINERAIS POR MEIO DENSO: UMA REVISÃO DA LITERATURA*, Monografia, Universidade Federal de Goiás – UFG, 2015

AUSTIN, L. G.; LUCKIE, P. T., *Estimation of non-normalized breakage distribution parameters from batch grinding*. Powder Technology 5 (5), 267–277, 1972

AUSTIN, L. G.; KLIMPEL, R. R.; LUCKIE, P. T. *Process engineering of size reduction: ball milling*, Society of Mining Engineers of the American Institute of Mining, Metallurgical, and Petroleum Engineers. Inc., New York, p. 22-28, 1984.

BARNES, J. G. P. *An algorithm for solving non-linear equations based on the secant method*. The Computer Journal, v. 8, n. 1, p. 66-72, 1965.

BROYDEN, C. G. *A class of methods for solving nonlinear simultaneous equations*. Mathematics of computation, v. 19, n. 92, p. 577-593, 1965.

CAMPOS, T. M., *Modelagem Matemática da Prensa de Rolos Aplicada à Cominuição de Minério de Ferro*. Projeto final de curso, UFRJ, Rio de Janeiro: COPPE/UFRJ, 2018.

CARVALHO, R.M. *Modelagem, Simulação e Controle da Moagem a Seco em Moinho de Bolas*. Projeto final de curso, DMM, UFRJ, Rio de Janeiro: COPPE/UFRJ, 2007.

DOBBY, G. S.; FINCH, J. A., *An empirical model of capture in a high gradient magnetic separator and its use in performance prediction*. Society of Mining engineers of AIME Annual Meeting Atlanta, Georgia1977.

DUTRA, R. *Beneficiamento de Minerais Industriais*., II Encontro de Engenharia e Tecnologia dos Campos Gerais. [s.l.: s.n.], 2008.

FLANAGAN, D., *JavaScript: the definitive guide*. O'Reilly Media, Inc., 2006.

GONZAGA, L. M., *Separação Magnética a Úmido de Minérios de Ferro Itabiríticos*. Tese de Doutorado. Dissertação de Mestrado, Programa de Engenharia Metalúrgica e de Materiais, COPPE, Universidade Federal do Rio de Janeiro, 2014.

GUO, P., *Python Is Now the Most Popular Introductory Teaching Language at Top U.S. Universities, 2014.* Disponível em: <https://cacm.acm.org/blogs/blog-cacm/176450-python-is-now-the-most-popular-introductory-teaching-language-at-top-u-s-universities>. Acesso em 14 set. 2018, 07:18

GUPTA, A.; YAN, D. S., *Mineral processing design and operations: an introduction*. Elsevier, 2016.

HARTMAN, H. L.; MUTMANSKY, J. M., *Introductory mining engineering*. John Wiley & Sons, 2002.

HERBST, J. A.; FUERSTENAU, D. W., *Scale-up procedure for continuous grinding mill design using population balance models*. International Journal of Mineral Processing 7, 1–31, 1980.

KLIMA, M. S.; KIM, B. H., *Dense-medium separation of heavy-metal particles from soil using a wide-angle hydrocyclone*, Journal of Environmental Science and Health, Part A, v. 33, n. 7, pp. 1325-1340, 1998

JUNIOR, R. F. M., *Implementação de Middleware baseada em Microserviços*. Projeto final de curso. Universidade Federal de Pernambuco, 2015

LUZ, A. B. da; SAMPAIO, J. A.; FRANÇA, S. C. A., *Tratamento de Minérios*, 5. ed. Rio de Janeiro: CETEM/MCT, 2010. 896 p., 2010

LYNCH, A. J.; NAPIER-MUNN, T. J., *The modelling and computer simulation of mineral treatment processes—current status and future trends*. Minerals Engineering, v. 5, n. 2, p. 143-167, 1992.

MARTI, S. K.; ERDAL, F. M.; SHOHAM, O.; SHIRAZI, S. A., *Analysis of Gas Carry-Under in Gas-Liquid Cylindrical Cyclones*, The University of Tulsa and G. E. Kouba Chevron Petroleum Technology Company, 1996

MATHWORKS, *MATLAB API for Python*. 2018. Disponível em: <https://www.mathworks.com/help/matlab/matlab-engine-for-python.html>. Acesso em 14 set. 2018, 05:47

MARTINS, L. *Produção de concentrado de zinco a partir de minério silicatado com redução no teor de carbonatos*. Tese de Doutorado – UFMG. 2011

NARASIMHA, M.; BRENNAN, M.; HOLTHAM, P.N., *A Review of CFD Modeling for Performance Predictions of Hydrocyclone*. Engineering Applications of Computational Fluid Mechanics, vol. 1, No. 2, pp. 109-125, 2007.

ORBACH, O.; CROWE, C.M., *Convergence promotion in the simulation of chemical processes with recycle -- the dominant eigenvalue method*. Can. J. Chem. Eng., 49: 509--513, 1971.

PERLINGEIRO, C.A.G. *Engenharia de Processos*. Rio de Janeiro: Edgard Blucher, 2005

ROSEN, *Edward M. A review of quasi-Newton methods in nonlinear equation solving and unconstrained optimization*. In: Proceedings of the 1966 21st national conference. ACM, p. 37-41, 1966.

PYTHON, *Schools using Python*. 2017. Disponível em: <https://wiki.python.org/moin/SchoolsUsingPython>. Acesso em 14 set. 2018, 07:17

SAMPAIO, C. H.; TAVARES, L. M. M., *Beneficiamento Gravimétrico*. Editora UFRGS, 2005

SILVA, A. T., *Otimização da Moagem de escória granulada de alto-forno por meio da simulação computacional usando o modelo do balanço populacional*. Dissertação de mestrado, Rio de Janeiro: COPPE/UFRJ, 2007.

SLATER, J. W., *Examining Iterative Convergence*, 2008

SMYTH, I.C.; THEW, M.T., *A study of the effect of dissolved gas on the operation of liquid-liquid hydrocyclones*, Proceedings of Hydrocyclones '96 Conference. MEP. pp. 357-368, 1996

SOUZA, Maria Luiza, *Separação por Meio Denso*. Universidade De La Republica – Uruguai e UFRGS – Brasil. Montevideo e Porto Alegre, [s.n.], ago. 2018. Transparência.

SVAROVSKY, L., *Solid-Liquid Separation*, 3a edição, Butterworths, London, 1990.

TAGGART, A.F., *Handbook of Mineral Dressing*. New York: John Wiley, 1945.

WASSON, M.; CELARIER, S., *Microservices arquitecture style*. Documentação Microsoft. Disponível em: <https://docs.microsoft.com/en-us/azure/architecture/guide/architecture-styles/microservices>. Acesso em 14 set. 2018, 03:10.

WEGSTEIN, J. H. *Accelerating convergence of iterative processes*. Communications of the ACM, v. 1, n. 6, p. 9-13, 1958.

WHITEN, W. J., *Ball mill simulation using small calculators*. Proceedings, Australasian Institute of Mining and Metallurgy, p. 47-53, 1976

WHITEN, W. J., *The Simulation of Crushing Plants with Models Developed using Multiple Spline Regression.* Journal of the South African Institute of Mining and Metallurgy, 1972: 257-264.

W3C, *HTML 5.2 W3C Recomendation 14 December 2017*. Disponível em: <https://www.w3.org/TR/html/>. Acesso em 14 set. 2018, 04:00