Pontifícia
Universidade
Católica
do Rio de Janeiro

Projeto Final de Programação

David Beyda
dbeyda@inf.puc-rio.br

Agosto 2021

1 Introdução

A indústria de anúncios online (*online advertising*) é um dos setores de tecnologia que mais cresceram e continuam a crescer nos ultimos anos. Durante o ano de 2020, apesar da pandemia do corona-vírus, o mercado de anúncios online cresceu mais de 12%, atingindo uma receita global de 139,8 bilhões de dólares [1]. Além disso, a pandemia impulsionou mais comerciantes para a venda digital (*ecommerce*), tornando a consolidação de uma presença *online* algo vital para o sucesso desses comerciantes. Paralelo a isso, tem-se a adoção da tecnologia 5G, que possibilita novos formatos de anúncio, e em mais dispositivos.

Nesse contexto, empresas que atuam como intermediadores de anúncios online, como Google, Facebook, Microsoft, Yahoo, por exemplo, possuem grande incentivo para tentarem otimizar o sistema. Isso acontece porque a renda de um intermediador de anúncios online está diretamente ligada com sua eficiência. Um sistema mais eficiente na alocação de anúncios gera custos por ação (*cost-per-click*, *cost-per-purchase*) mais baixos para o anunciante, trazendo mais anunciantes para a plataforma, e dando espaço para esses anunciantes investirem mais. Do ponto de vista do anunciante, ele consegue uma operação de negócio mais eficiente, seus custos reduzidos lhe dão mais espaço para crescer, e a viabilidade do negócio é atingida mais facilmente.

Esse ambiente motivou o estudo teórico dos problemas chamados de *Online Matching* e suas generalizações, que são aplicados na prática [2] e geram impacto direto no negócio dessas empresas. O problema estudado por esse trabalho é

chamado de *Online Packing*, e pode ser aplicado ao leilão de anúncios pelo lado do anunciante. Basicamente, o anunciante define o orçamento disponível por dia, e ele precisa decidir quais espaços de propaganda comprar, conforme eles ficam disponíveis, em tempo real, sem ultrapassar o orçamento diário estabelecido. É claro que o anunciante em si não faz isso manualmente. Geralmente a plataforma intermediadora dos anúncios disponibiliza uma ferramenta que encapsula esse algoritmo para o anunciante.

Mais formalmente, o modelo de *Online Packing* é o seguinte: o problema se desenrola em T instantes de tempo. A cada instante, o usuário (ou algorítmo) recebe n items, cada item com seu valor de recompensa e seu vetor de custos. O usuário então decide se pega algum item, e qual item pega, recebendo sua recompensa e incorrendo em seus custos. Depois de feita a escolha, o usuário avança para o próximo instante de tempo e recebe novamente n items, até que se chegue ao instante T. Além disso, o usuário tem uma restrição sobre os custos, que devem ser menores ou iguais a B em cada dimensão de custo. O objetivo do problema é maximizar a soma dos valores do itens escolhidos. Para uma apresentação mais detalhada do problema e do modelo no qual ele opera (suas premissas e suposições), consulte a Documenação do Usuário [1].

Portanto, esse trabalho é uma implementação de um algorítmo de *Online Packing* proposto por Agrawal e Devanur [2]. Com esse projeto, busca-se facilitar o estudo desse tipo de problema e entender melhor quais dificuldades de implementação aparecem na prática. Além disso, o componente que modela o problema de *Online Packing* foi desenvolvido de forma modular e desacoplada, permitindo que seja reutilizado por outras implementações de algorítmos para o problema de *Online Packing*.

2 Especificação

Neste capítulo será apresentada a especificação do projeto, por meio do escopo, dos requisitos funcionais e dos não-funcionais.

2.1 Escopo

O projeto visa apresentar uma implementação clara, simples e objetiva do algorítmo proposto por Agrawal e Devanur (algorítmo 6.1 em [2]) para o problema de *Online Packing*. Foge do escopo desse trabalho a implementação de um *solver* de MIP (*Mixed Integer Programming*) a ser usado pelo algorítmo. Portanto, para essa finalidade, é razoável que se utilize um *solver* externo, de preferência, um bem conhecido e documentado.

2.2 Requisitos Não-Funcionais

Aqui são listados os requisitos não-funcionais especificados para o projeto, bem como sua justificativa. São listados a seguir:

[1] https://dbeyda.github.io/fast-online-packing/

- **Usabilidade**: na medida em que o projeto tem como objetivo facilitar o estudo, ele deve ser de fácil aprendizado e uso.
- **Manutenibilidade**: relacionado com a facilidade de uso e compreensão, o projeto deve requerir baixo esforço para ser mantido, buscando a simplicidade e objetividade ao invés de tentar englobar ou prever todas as possíveis possibilidades para um problema.
- **Portabilidade**: novamente devido à finalidade do projeto, ele deve ser portátil e fácil de ser executado e modificado em um computador com qualquer sistema operacional. Execução em dispositíveis móveis, como celulares, não se fazem necessárias.
- **Reusabilidade**: é ideal que parte desse projeto possa ser reutilizado ou até evoluído para auxiliar na implementação de outros algorítmos para o problema.

2.3 Requisitos Funcionais e Casos de Uso

Nesta seção, descrevemos os principais requisitos funcionais do projeto. Os requisitos do ponto de vista do usuário que pretende utilizar o algoritmo implementado são apresentados na tabela 1. Os requisitos para um usuário que pretende desenvolver um novo algoritmo para o mesmo problema são apresentados na tabela 2.

ID	**Caso de Uso**	**Descrição**
UC1	Gerar uma instância de testes	O usuário deve conseguir gerar uma instância aleatória do problema que seja válida para o algorítmo.
UC2	Utilizar o algoritmo para saber qual item escolher	O usuário deve poder utilizar o algoritmo implementado, apresentando as opções e recebendo, a cada round, a escolha do algoritmo.
UC3	Obter a solução ótima	O desenvolvedor deve ser capaz de obter a solução ótima *offline* da instância do problema.
UC4	Ver resultado do experimento	O usuário pode obter um relatório com detalhes da execução e do resultado obtido.

Tabela 1: Casos de Uso de usuário final

3 Projeto

Nesta seção serão apresentados detalhes da implementação do projeto, os problemas encontrados, as soluções escolhidas e outras escolhas de projeto feitas

ID	Caso de Uso	Descrição
UC5	Apresentar as opções de itens disponíveis no instante atual	O desenvolvedor deve poder registrar na instância do problema as opções de items disponíveis no momento corrente.
UC6	Verificar se uma opção de item pode ser escolhida	A instância do problema deve permitir a fácil verificação de quais items podem ser escolhidos sem violar as restrições.
UC7	Registrar a opção de item escolhida	O desenvolvedor deve poder registrar a opção escolhida em cada instante na instância do problema.

Tabela 2: Casos de Uso ao desenvolver um novo algoritmo

durante o planejamento do projeto.

O projeto foi concebido como uma biblioteca de Python[2], a ser distribuída pelo PyPI[3], o repositório oficial de pacotes para Python. A linguagem Python foi escolhida pelas seguintes características, que se alinham com o desenvolvimento do projeto:

- Alta adoção da linguagem (reusabilidade).
- Facilidade de aprendizado e uso da linguagem (usabilidade, manutenibilidade).
- É uma linguagem interpretada, e não compilada, o que facilita sua distribuição e uso (portabilidade).

Além disso, a escolha por fazer o projeto no formato de uma biblioteca, assim como por distribuí-la pelo PyPI se deu com o objetivo de facilitar a instalação e utilização do projeto pelo usuário final.

Com relação aos detalhes de implementação, optou-se por fazer um código limpo e explicativo, mesmo que se perca em velocidade de execução. Tais escolhas foram tomadas buscando resgatar o objetivo do projeto, que não é ser usado em ambientes de produção, mas sim em ambientes de prototipagem e estudos.

3.1 Arquitetura

Aqui serão abordados os aspectos arquiteturais do projeto da biblioteca.

3.1.1 Modelagem da Instância do Problema

Os seguintes dados definem uma instância do problema de *Online Packing*:

- Capacidade em cada dimensão (B: número).

[2]https://www.python.org/
[3]Python Package Index - https://pypi.org/

- Tamanho do vetor de custos, ou dimensão dos custos (d: número inteiro).
- Número de instantes de tempo (T: número inteiro).
- Valor dos itens disponíveis no tempo t (V_t: lista de números).
- Custo dos itens disponíveis no tempo t (C_t: lista de vetores de custo).

Por exemplo, considere que os itens disponíveis sejam os seguintes, para um determinado instante t:

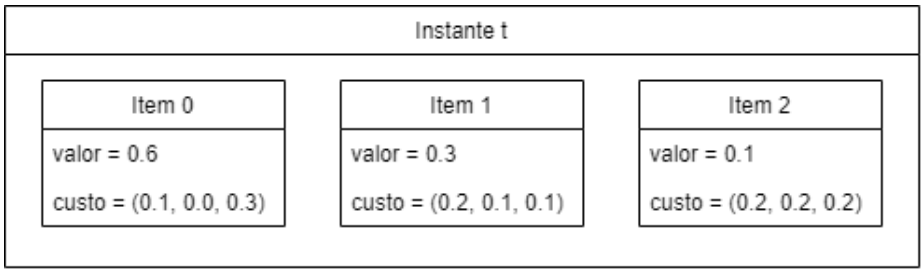

Figura 1: Exemplo de itens disponíveis no instante t

As opções de itens disponíveis nesse instante serão representadas por dois vetores: V_t (vetor dos valores) e C_t (vetor dos custos). A representação vetorial será a seguinte:

```
V_t = [0.6, 0.3, 0.1]
C_t = [
        [0.1, 0.0, 0.3],
        [0.2, 0.1, 0.1],
        [0.2, 0.2, 0.2]
       ]
```

Essa representação para o instante do problema busca a praticidade e simplicidade de uso. A escolha feita pelo algorítmo será representada apenas pelo índice do item escolhido (no exemplo, pode ser 0, 1, ou 2), e a escolha de não pegar nenhum item será representada pelo índice -1.

Além disso, enquanto B, d, T são fornecidos antes do início do problema, V_t e C_t são fornecidos apenas no instante t, resultando na característica *online* do problema.

3.1.2 Visão de Alto Nível dos Módulos

Nessa seção, apresentamos uma visão de alto nível de como a biblioteca será organizada. A bibiloteca será composta de 6 módulos. Alguns deles serão apenas auxiliares, e alguns poderão ser usados individualmente e independentemente. Mais detalhes sobre a implementação e API de entrada e saída podem ser encontrados na Documentação para o Usuário [4].

A seguir, apresentamos o diagrama UML dos módulos concebidos, e em seguida, apresentamos suas respectivas funções.

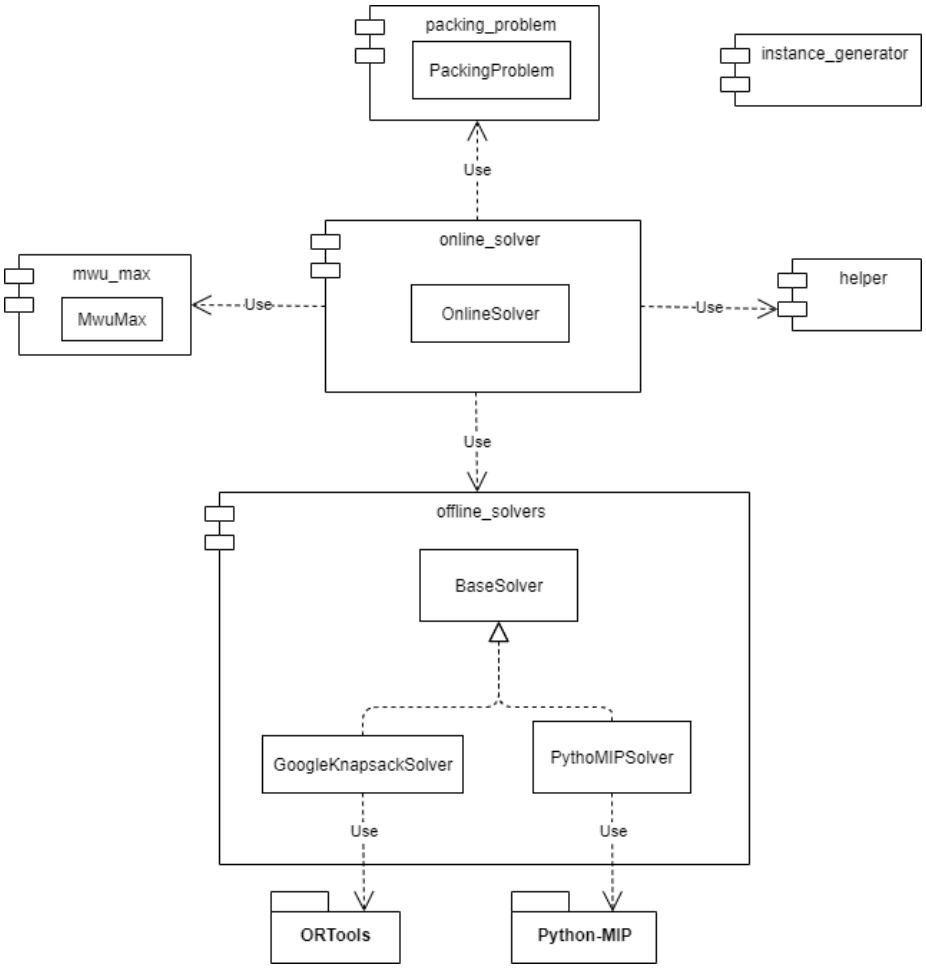

Figura 2: Diagrama UML de alto nível da biblioteca

[4] https://dbeyda.github.io/fast-online-packing/

Instance Generator Module. Esse módulo gera instâncias para o problema. Ele é capaz de gerar dois tipos de instâncias: instâncias aleatórias e instâncias com garantais (que respeitam as restrições do problema). Para ambas, os valores e custos dos itens são gerados como números aleatórios no intervalo $[0, 1)$. Para instâncias aleatórias, a capacidade B é gerada também aleatoriamente. Para instâncias com garantias, a capacidade gerada é a menor possível para que as garantias sejam válidas. Ela é encontrada por meio de uma busca binária sobre os números decimais. Esse módulo é usado nos testes, e pode ser usado para simulações com o algorítmo.

Helper. Esse módulo contém funções extras que não fazem parte do domínio de nenhum módulo em especial, como funções matemáticas para cálculo com vetores. É usado pelo *Online Solver Module*.

MWU Max Module. Esse módulo implementa um algoritmo de MWU (Multiplicative Weights Update) que busca maximizar uma função. Conforme descrito em [2], esse MWU é adaptado para ter um *expert* extra, para ter maior flexibilidade na alocação de seus pesos. Esse módulo é usado pelo *Online Solver Module*.

Packing Problem Module. Esse módulo encapsula e aplica todas as regras e características do problema de *Online Packing*, podendo ser usado de forma independente do resto dessa bibiloteca. Ele é usado pelo *Online Learning Module* para garantir e encapsular o cumprimento das regras do problema.

Offline Solver Module. Esse módulo contém dois *solvers* que resolvem o problema de *Packing* de forma *offline*, seguindo uma classe abstrata comum. Escolheu-se por implementar dois *solvers* por dois motivos: assegurar corretude na implementação de ambos comparando seus resultados, e demonstrar como a biblioteca pode ser estendida para o uso de novos *solvers*, que também devem herdar da classe abstrata. Esses *solvers* são usados pelo *Online Packing Module* para encontrar um parâmetro do problema (Z), e também para encontrar o resultado ótimo ao final do jogo, para fins de avaliação do algoritmo.

Online Solver Module. É o módulo principal da biblioteca, que implementa o algoritmo de *Online Packing* descrito por Agrawal & Devanur [2].

4 Testes

O projeto utiliza testes unitários para assegurar a corretude dos módulos e auxiliar no desenvolvimento. Utilizou-se a biblioteca PyTest[5] para facilitar o desenvolvimento dos testes, devido a possibilidade de agrupar os testes em classes, criando diferentes suítes de testes. Todos os módulos foram testados, cada

[5] https://pytest.org/

um com sua própria suíte de testes. Os testes estão no diretório "*tests*", na raíz do projeto.

Na raíz do projeto, o arquivo *README.md* contém as instruções para executar os testes, bem como para salvar o resultado dos testes em um arquivo. É recomendado que antes de qualquer alteração no código, se faça a alteração correspondente nos testes primeiro, e depois se altere o código para fazer o teste ser aceito, conforme os princípios de TDD (*Test Driven Development*).

Os resultados dos testes unitários realizados serão apresentados conforme a Figura 3. Caso haja falha em algum teste, o teste falhou será indicado e o trecho do código que gerou a falha também.

```
===================== test session starts =====================
platform linux -- Python 3.9.4+, pytest-6.2.3, py-1.10.0,
pluggy-0.13.1
rootdir: /mnt/c/dev/agrawal-devanur-online-packing
collected 30 items

tests/unit_tests/test_helper.py ...                          [ 10%]
tests/unit_tests/test_instance_generator.py .....            [ 26%]
tests/unit_tests/test_mwu_max.py .......                     [ 50%]
tests/unit_tests/test_offline_solvers.py ..                  [ 56%]
tests/unit_tests/test_online_solver.py ....                  [ 70%]
tests/unit_tests/test_packing_problem.py .........           [100%]

====================== 30 passed in 5.14s ======================
```

Figura 3: log da execução dos testes.

5 Instalação do projeto

A biblioteca está disponibilizada no PyPI (Python Package Index) sob o título de "*fast-online-packing*" [6], em referência ao título do artigo que originou o algorítmo. Além disso, o código da biblioteca está disponível no GitHub [7].

Para instalar o projeto, é necessário usar Python 3.9 ou acima, e basta instalar a biblioteca. O arquivo *README.md*, na raíz do projeto, contém um passo a passo com os comandos para instalar a bibiloteca em um sistema *Ubuntu*.

[6] https://pypi.org/project/fast-online-packing/
[7] https://github.com/dbeyda/fast-online-packing

6 Documentação para o Usuário

6.1 Ferramentas e Infraestrutura

A ferramenta escolhida para facilitar a documentação foi o Sphinx [8]. Ele permite que um desenvolvedor escreva a documentação em diversas páginas no formato *reStructuredText* (.rst), que são agrupados e resultam em um site estático em HTML. Além disso, utilizou-se a extensão *autodoc* [9], que permite gerar uma referência de API (*Application Programming Interface*) a partir das *docstrings* utilizadas para documentar o código Python. Além disso, há uma outra parte da documentação que se refere aos tutoriais, exemplos de uso e artigos didáticos. Essa parte da documentação também foi gerada com o Sphinx, porém de forma manual.

A documentação para o usuário está hospedada no GitHub Pages [10], que nos permite hospedar um site estático. Mais ainda, como o repositório do projeto está no GitHub, podemos indicar para o GitHub Pages o diretório do projeto que contém o site estático em HTML, que deve ser servido. Assim, a cada atualização no repositório remoto, o site estático também é atualizado para a versão mais recente.

Os comandos para gerar a documentação podem ser encontrados no arquivo *README.md*, na raíz do projeto.

6.2 Links Relacionados

- Repositório no GitHub:
  https://github.com/dbeyda/fast-online-packing
- Projeto no PyPI:
  https://pypi.org/project/fast-online-packing/
- Documentação para o usuário:
  https://dbeyda.github.io/fast-online-packing/

[8] https://www.sphinx-doc.org/
[9] https://www.sphinx-doc.org/en/master/usage/extensions/autodoc.html
[10] https://pages.github.com/

7 Conclusões

Nesse relatório, foi apresentada a especificação e o projeto de uma bibiloteca que implementa um algoritmo para o problema de *Online Packing*. O projeto da biblioteca tinha como um dos objetivos ser implementado de formula modular, permitindo que seus módulos sejam usados de forma independente para auxiliar no desenvolvimento de outros algoritmos para o mesmo problema. Esse objetivo foi cumprido com sucesso.

Além disso, foram apresentadas as principais decisões de projeto em relação à sua especificação, projeto, testes e documentação. Por fim, espera-se que esse projeto possa servir como base para a implementação de algoritmos futuros, e que ajude a prototipagem de provas de conceitos para novos sistemas.

Referências

[1] Internet advertising revenue report: Full-year 2020 results, 2021.

[2] Shipra Agrawal and Nikhil R. Devanur. Fast algorithms for online stochastic convex programming. *CoRR*, abs/1410.7596, 2014.