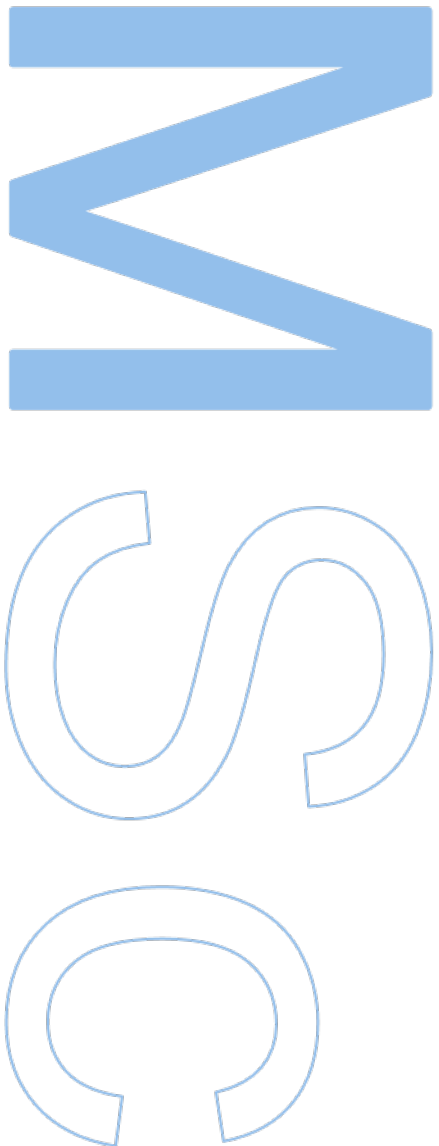
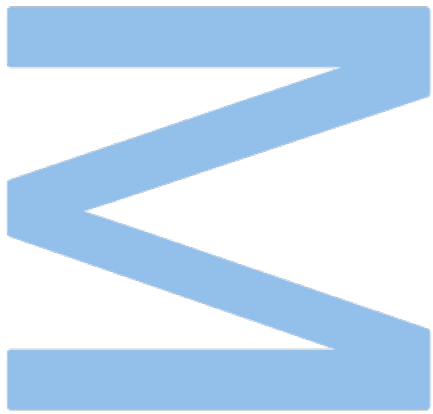
Modelação Matemática do Planeamento de Encomendas numa Fábrica Vertical de Plantas

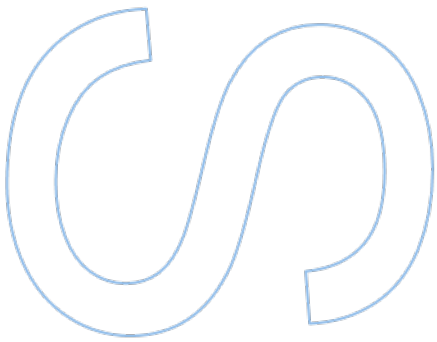
Jorge Miguel Barbosa Ferreira
Mestrado em Engenharia Matemática
Departamento de Matemática
2023

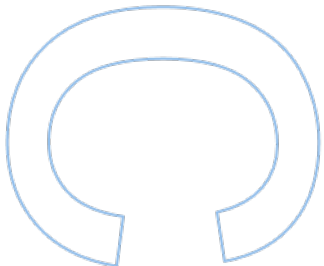
Orientador
Sílvio Marques de Almeida Gama, Professor Associado,
Departamento de Matemática, Faculdade de Ciências da Universidade do Porto




U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO







Declaração de Honra

Eu, Jorge Miguel Barbosa Ferreira, inscrito(a) no Mestrado em Engenharia Matemática da Faculdade de Ciências da Universidade do Porto declaro, nos termos do disposto na alínea a) do artigo 14.ž do Código Ético de Conduta Académica da U.Porto, que o conteúdo da presente dissertação reflete as perspetivas, o trabalho de investigação e as minhas interpretações no momento da sua entrega.

Ao entregar esta dissertação, declaro, ainda, que a mesma é resultado do meu próprio trabalho de investigação e contém contributos que não foram utilizados previamente noutros trabalhos apresentados a esta ou outra instituição.

Mais declaro que todas as referências a outros autores respeitam escrupulosamente as regras da atribuição, encontrando-se devidamente citadas no corpo do texto e identificadas na secção de referências bibliográficas. Não são divulgados na presente dissertação quaisquer conteúdos cuja reprodução esteja vedada por direitos de autor.

Tenho consciência de que a prática de plágio e auto-plágio constitui um ilícito académico.

Jorge Miguel Barbosa Ferreira

Porto, 25 de julho de 2023

Agradecimentos

Ao meu orientador, Professor Sílvio Gama, agradeço-lhe pela dedicação, disponibilidade, paciência e carinho em todos os momentos.

Aos meus colegas e amigos, Pedro, Carla e Fernando, pela paciência, carinho e disponibilidade em todos os momentos que precisei e que eles precisaram e que serão para toda a vida.

A todos os professores que me incentivaram e marcaram académica e pessoalmente.

A uma pessoa muito especial, Ana Cristina, que me ajudou em muitos momentos e fez-me ver que às vezes também tenho de me pôr em primeiro lugar. Uma amizade que durará para a vida.

A todas as pessoas que permitiram que estivesse apto a seguir os meus sonhos e a encarar os desafios com um sorriso.

À minha família, em especial, à minha mãe e à minha irmã por estarem sempre ao meu lado e por acreditarem em mim. Agradeço também aos meus amigos patudos pela companhia e carinho com que me recebiam quando chegava a casa e me levantavam o ânimo.

Este meu trabalho beneficiou da Bolsa de Investigação para Licenciado, no âmbito do projeto SNAP - Sustainable Management and Control of Agro-Production Systems, com a referência NORTE-01-0145-FEDER-000085, financiado Fundo Europeu de Desenvolvimento Regional (FEDER), através do Programa Operacional Regional Norte de Portugal (NORTE2020), no âmbito do Acordo de Parceria PORTUGAL 2020.

Resumo

A agricultura é uma atividade essencial na produção de fruta e legumes que tem sofrido grandes alterações no seu método e processo de cultivo ao longo do tempo.

Com o crescente aumento da população, as alterações climáticas e o requisito para um desenvolvimento sustentável, surge um novo método de produção de alimento vegetal, a agricultura vertical. Este tipo de agricultura caracteriza-se por produzir alimentos recorrendo a pouco/nenhum uso de pesticidas e herbicidas (dado estar-se num ambiente fechado e controlado), usar fontes de energia renováveis, reduzir distâncias de transporte e pegada de carbono (dado as fábricas verticais poderem ser construídas na periferia urbana).

Analogamente a outros métodos de produção, a agricultura vertical está também sujeita a problemas de gestão e planeamento de produção que requerem uma análise cuidadosa e extensiva, em particular minimizar custos, tempo de trabalho, gestão do movimento de paletes dentro da fábrica, etc.

Esta dissertação explora um modelo matemático que visa minimizar o número de movimentos de paletes numa fábrica vertical de plantas. Um dos modelos que pode ser usado nesta situação é o descrito em Santini et al. 2021. Assim sendo, nesta dissertação, vamos explorar o modelo de programação linear CGPP – *Crop Growth Planning Problem* – que otimiza o movimento de paletes numa fábrica vertical de crescimento de plantas.

O modelo de programação linear que aqui desenvolvemos, foi implementado em *Python* com recurso, em particular, ao pacote de *software* de otimização *Pyomo*. Apresentamos também várias simulações computacionais correspondentes a diversas configurações da carteira de encomendas, número de paletes utilizadas, colheitas de diferentes produtos alimentares, etc.

Palavras-chave: agricultura vertical, fábrica de plantas, modelos matemáticos de planeamento do crescimento das plantas, CGPP, Pyomo.

Abstract

Agriculture is an essential activity in the production of fruit and vegetables that has undergone major changes in its method and process of cultivation over time.

With the increasing population, climate change and the requirement for sustainable development, a new method of plant food production emerges, vertical farming. This type of agriculture is characterized by producing food using little/no use of pesticides and herbicides (since it is in a closed and controlled environment), using renewable energy sources, reducing transport distances and carbon footprint (since vertical factories can be built on the urban periphery).

Similarly, to other production methods, vertical farming is also subject to management and production planning problems that require careful and extensive analysis, in particular minimizing costs, working time, managing the movement of pallets within the factory, etc.

This dissertation explores a mathematical model that aims to minimize the number of pallet movements in a vertical plant factory. One of the models that can be used in this situation is the one described in Santini et al. 2021. Therefore, in this dissertation, we will explore the CGPP linear programming model – *Crop Growth Planning Problem* – that optimizes pallet movement in a vertical plant growth factory.

The linear programming model that we developed here was implemented in *Python* using the *software* optimization package *Pyomo*. We also present several computer simulations corresponding to various demand configurations, number of pallets used, harvests of different food products, etc.

Key-words: vertical agriculture, plant factory, mathematical models on planning the plant growth, CGPP, Pyomo.

Índice

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

A agricultura é uma das atividades mais importantes para a produção de alimento vegetal e de matérias-primas, e constituiu a chave fundamental para o desenvolvimento de civilizações sedentárias.

O crescimento da população e a necessidade de a alimentar levou ao progresso de técnicas da agricultura de modo a produzir maior quantidade de alimento.

No entanto, a agricultura tem estado sujeita a novos desafios relacionados com a área de cultivo e sustentabilidade. Dada a natureza desses desafios, um novo método de produção vegetal – agricultura vertical – que pode ajudar a solucioná-los tem despertado interesse recentemente.

Com a produção de plantas em larga escala, e, em geral com a indústria, estão associados problemas e questões pertinentes, nomeadamente qual a melhor localização, quais os métodos de produção, de que forma podem ser otimizados os recursos, e o seu impacto na sociedade.

A tomada de decisões para algumas dos problemas mencionados é ajudada com recurso a modelos e ferramentas matemáticas, por exemplo, os modelos de planeamento de tarefas que podem ser usados na organização da produção, e que têm aplicações nas mais diversas áreas.

Dos vários modelos que existem, vamos focar-nos no modelo CGPP - *Crop Growth Planning Problem* - de planeamento da produção em fábricas de plantas em regime de agricultura vertical.

O contributo desta dissertação é explorar o modelo CGPP com vista à minimização do movimento de paletes de plantas ao longo das suas diferentes fases de crescimento. Em

particular, apresentamos também um caso/modelo simplificado que ilustra, de forma simples, o significado e a interpretação da notação utilizada para as variáveis de decisão e as restrições a que estão sujeitas.

A dissertação está organizada da seguinte forma. No segundo capítulo, faremos uma breve introdução à agricultura vertical, referindo os principais objetivos, as vantagens e desvantagens.

No terceiro capítulo, faremos uma breve revisão bibliográfica sobre modelos de produção de plantas, introduziremos a notação, as variáveis e os problemas que estão associados.

O modelo final, as simplificações tomadas e os resultados de simulações computacionais são feitos no quarto capítulo. Terminámos a dissertação com as conclusões e possíveis trabalhos futuros.

Capítulo 2

Agricultura Vertical: revisão bibliográfica

Com o decorrer do tempo, a produção agrícola tem vivenciado grandes desafios referentes à área e capacidade de cultivo e à preservação dos recursos naturais que são finitos.

Além disso, com a tendência crescente do aumento da população e urbanização e com as mudanças climáticas, a área arável por pessoa tem diminuído.

Nesse sentido, têm sido criados sistemas de produção agrícola e sustentável capazes de satisfazer as necessidades da procura, minimizando o espaço usado, a distância a centros urbanos e o impacto ambiental (Objetivos de Desenvolvimento Sustentável, ONU). O conceito de agricultura vertical tem ganho mais adeptos.

Agricultura vertical é um método de produção agrícola feita, geralmente, em ambiente fechado, controlado e sustentável, onde é preferida a produção vertical numa área reduzida em vez da produção em extensão (isto é, usando o máximo de área horizontal disponível), e é usado em fábricas de plantas.

Uma fábrica de plantas é um complexo de produção de plantas com uma estrutura termicamente isolada e hermética. Através destas fábricas, é mais fácil a gestão de grandes áreas de produção.

A produção de plantas pode ser realizada em várias modalidades:

- Hidroponia (figura 2.1): método de crescimento de plantas usando uma solução aquosa de nutrientes minerais sem recurso a solo (Kozai, Niu e Takagaki 2019);

(a) Sistema hidropónico (Fonte: Pixabay.com)

(b) Ausência de solo (Fonte: Pixabay.com)

Figura 2.1: Hidroponia.

- Aquaponia (figura 2.2): método de crescimento de plantas sem recurso a solo em regime de piscicultura, isto é, usando a recirculação da água dos tanques de peixes para alimentar as plantas (Naskali, Pinarer e Tolga 2021, e Kozai, Niu e Takagaki 2019);

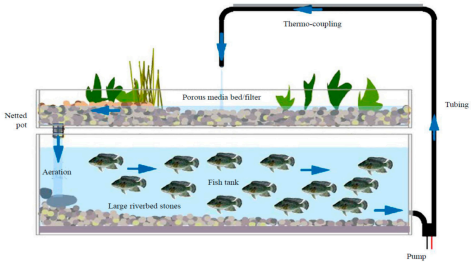


Figura 2.2: Sistema aquapónico. (Fonte: Al-Kodmany 2018)

- Aeroponia (figura 2.3): método de crescimento de plantas sem recurso a solo, onde as raízes das plantas são pulverizadas com uma solução aquosa (rica em nutrientes) em intervalos regulares (Naskali, Pinarer e Tolga 2021):

(a) Sistema aeropónico (Fonte: Pixabay.com)

(b) *Green Wall* (Fonte: Pixabay.com)

Figura 2.3: Aeroponia.

Estes métodos promovem o crescimento rápido das plantas, não usam fertilizantes e pesticidas, e reduzem e até eliminam o cultivo em solo. São processos limpos no sentido

em que não são usados excrementos de animais - que podem conter micro-organismos nefastos para as plantas.

A produção de plantas requer um estudo exaustivo da fisiologia e das condições de crescimento e maturação – quantidade de água, luz, nutrientes e dióxido de carbono, etc. – bem como das características do solo, da qualidade das sementes e, do ponto de vista da produção, da gestão de risco (Despommier 2013).

As fábricas de plantas podem ser desenhadas com o intuito de usar simultaneamente fatores naturais externos (luz solar, dióxido de carbono, água) e fatores gerados artificialmente, ou somente fatores artificiais. Naturalmente, existem diferenças estruturais, mas principalmente, no custo e rendimento da fábrica (Brandon et al. 2016).

A construção destas estruturas deve ter em conta fatores associados ao consumo energético, número de tipos de plantas produzidas, volume de produção, custos de transporte, risco e mão de obra (Benke e Tomkins 2017).

O uso de fatores naturais está sujeito a um controlo mais intensivo - o ar extraído do exterior contém dióxido de carbono e outros poluentes, assim como a água da chuva que, para além de conter outras substâncias, é mais ácida do que a água corrente. Deste modo, são necessários equipamentos mais sofisticados para o pré-tratamento destes fatores.

2.1 Crescimento das plantas: fatores e estruturas em fábricas

As plantas usam a luz solar para produzirem compostos orgânicos (nutrientes para crescimento) a partir de dióxido de carbono e água através de um processo designado por fotossíntese:

$$6\ \underbrace{CO_2}_{\text{dióxido de carbono}} + 12\ \underbrace{H_2O}_{\text{água}} \xrightarrow{\text{Energia luminosa}} \underbrace{C_6H_{12}O_6}_{\text{glicose}} + 6\ \underbrace{O_2}_{\text{oxigénio}} + 6\ \underbrace{H_2O}_{\text{água}} \tag{2.1}$$

A equação acima mostra que o dióxido de carbono, a água e a luz são os fatores que mais contribuem para a fotossíntese. De seguida, descrevemos a importância de cada um.

2.1.1 Dióxido de carbono e ventilação

Na construção de uma fábrica de plantas é requerida uma fonte de dióxido de carbono. Uma grande concentração deste gás resulta, geralmente, no aumento do rendimento da

produção, e a ventilação é usada para difundir CO_2 no ambiente de crescimento.

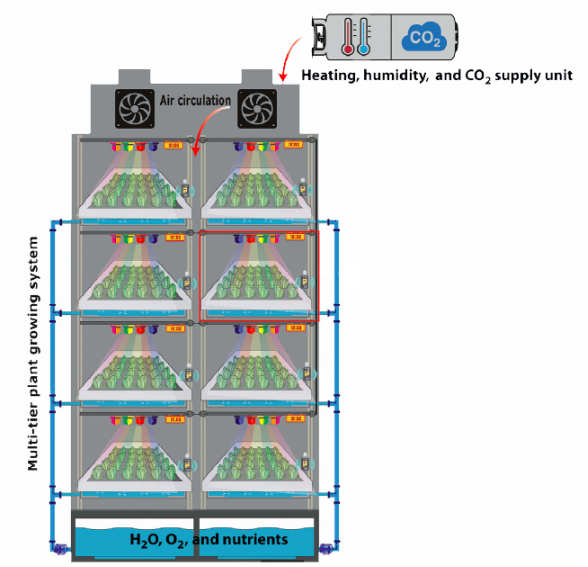


Figura 2.4: Inclusão de uma fonte de CO_2. (Fonte:SharathKumar, Heuvelink e Marcelis 2020)

No entanto, o sucesso do enriquecimento com CO_2 pode depender do clima, caso a fábrica seja construída com o intuito de usar os recursos naturais exteriores. Isto é, quando o clima exterior é ameno, a ventilação artificial pode ser desligada, e estará em contacto com o exterior - existindo a possibilidade da entrada de microrganismos nocivos -, o CO_2 pode ser injetado, contudo haverá difusão deste gás para o exterior. Quando a ventilação natural não é possível, existem estratégias de *enriquecimento de CO_2 de fluxo nulo*, onde é injetado este gás sempre que a sua concentração é inferior à do ambiente, minimizando as perdas de gás (Brandon et al. 2016).

2.1.2 Água

A água é essencial ao crescimento e desenvolvimento de todos os seres vivos, e tem um papel especialmente importante para as plantas.

A agricultura é uma das atividades económicas que mais consome água. Recentemente, temos assistido à diminuição da água potável acessível, à escassez em várias partes do globo, e ao seu uso exagerado e irresponsável. Deste modo, questionámo-nos se existem formas de usar água de forma mais eficiente possível sem que haja desperdício ou poluição.

Em qualquer um dos métodos de produção de plantas referidos no início deste capítulo é usada água - quer em estado líquido quer em estado gasoso (que concebe humidade

ao ambiente) -, e graças às estruturas existentes nas fábricas, a água encontra-se num circuito fechado, e é constantemente usada. Além disso, por se encontrarem num ambiente fechado e controlado, a água resultante da transpiração das plantas é colhida e pode ser usada neste circuito.

2.1.3 Luz e temperatura

A luz e a temperatura desempenham um papel importante no processo fotossintético, mas é a luz que mais influencia esse processo.

Através do estudo das condições de crescimento das plantas podemos modificar o seu ciclo de crescimento pela manipulação da luz. Isto é, modificando o fotoperíodo[1] em cada fase de crescimento podemos fazer crescer plantas num espaço de tempo inferior ao usual.

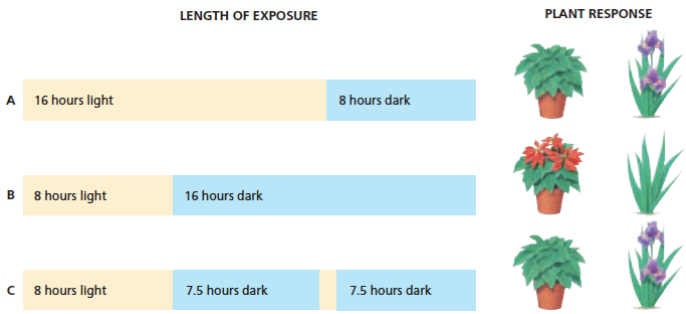


Figura 2.5: Resposta a diferentes períodos de luz e escuridão. (Fonte: Postlethwait e Hopson 2006)

O uso de luz natural tem benefícios energéticos para uma fábrica, quando esta se encontra numa zona com elevada exposição solar. Quando isso não acontece, é usada luz artificial.

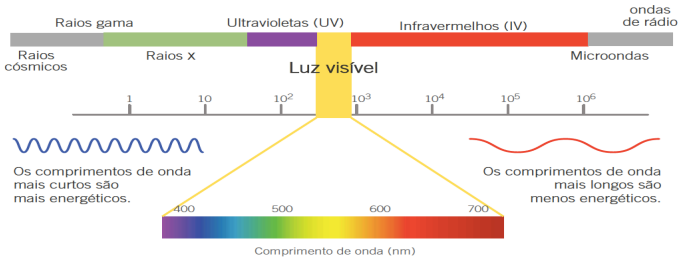


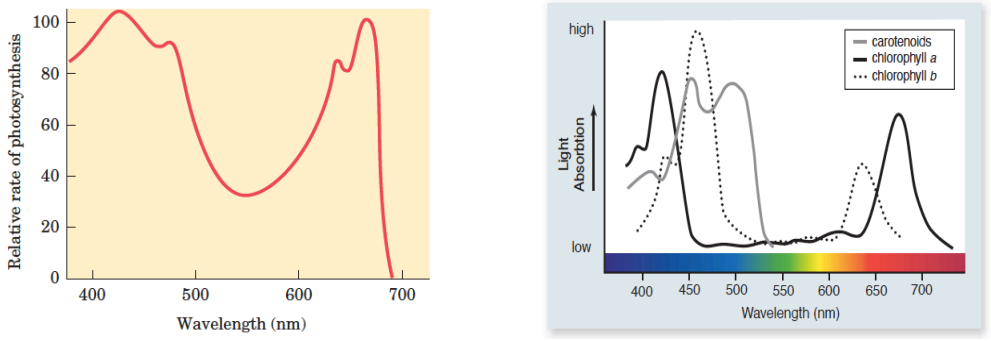
Figura 2.6: Espetro eletromagnético. (Fonte: Carrajola, Castro e Hilário 2007)

A luz solar tem o espetro característico indicado na figura 2.6 e não permite a sua manipulação.[2]

[1] fotoperíodo: tempo de exposição à luz necessário para que uma planta ou animal se desenvolvam naturalmente

[2] A porção do espetro visível pelo Homem está compreendida entre 400 e 700 nm.

Pela análise da figura (2.7) verifica-se que o processo fotossintético tem uma resposta desfavorável quando está sujeita a uma luz com comprimento de onda entre 500 e 600 nm (cor verde), uma vez que há menor absorção dos pigmentos clorofila-a, clorofila-b e carotenóides, importantes para a fotossíntese.



(a) Espetro de ação da fotossíntese (Fonte:Nelson, Nelson e Cox 2004)

(b) Espetro de absorção dos pigmentos existentes nos cloroplastos (organito celular) (Fonte: Johnson 2010)

Figura 2.7: Espetro de ação da fotossíntese e dos pigmentos.

O uso de vários tipos de lâmpadas como fontes de luz permite ajustar, adequadamente, o espetro de emissão para cada planta de forma que seja otimizada a produção das substâncias essenciais ao crescimento.

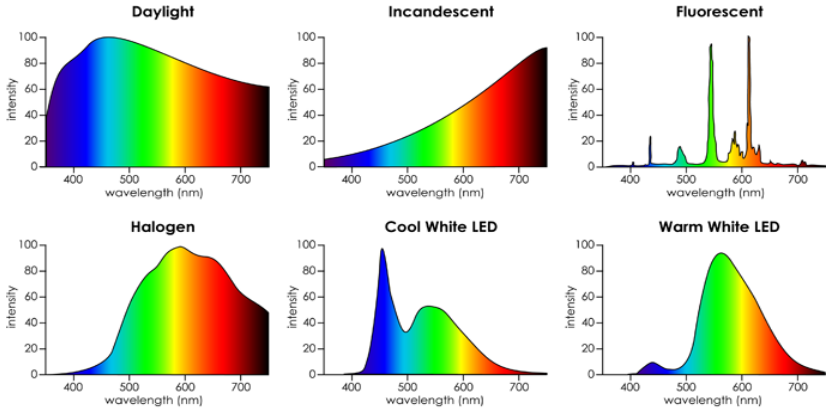


Figura 2.8: Espetro de várias lâmpadas (da esquerda para a direita): luz natural, incandescente, fluorescente, halogenada, LED branca (fria), LED branca (quente). (Fonte: lightingschool.eu)

A cada intervalo de comprimentos de onda da luz é possível notar que existem características das plantas que são favorecidas e outras que são inibidas (para o caso da planta da figura 2.9):

1. para comprimentos de onda entre 630 e 735 nm (luz vermelha equivalente a tempera-

turas mais amenas), as plantas são mais nutritivas (*biomass +*) e o comprimento e área das folhas é maior; por outro lado, à menor produção de pigmentos essenciais à fotossíntese (*chlorophyll -*);

2. para comprimentos de onda entre 440 e 535 nm (luzes azul e verde, associadas a temperaturas mais quentes), as plantas são mais nutritivas (*biomass +*), o comprimento e área das folhas é menor, mas a espessura é favorecida; a produção de pigmentos essenciais à fotossíntese é maior e verifica-se a produção de outras substâncias importantes (para as plantas e para os animais) (*chlorophyll +*);

3. para comprimentos de onda entre 580 e 600 nm e inferiores a 400 nm, as plantas tendem a perder o valor nutritivo e a produzirem substâncias (possivelmente) nocivas ao consumo humano (*nitrate +*).

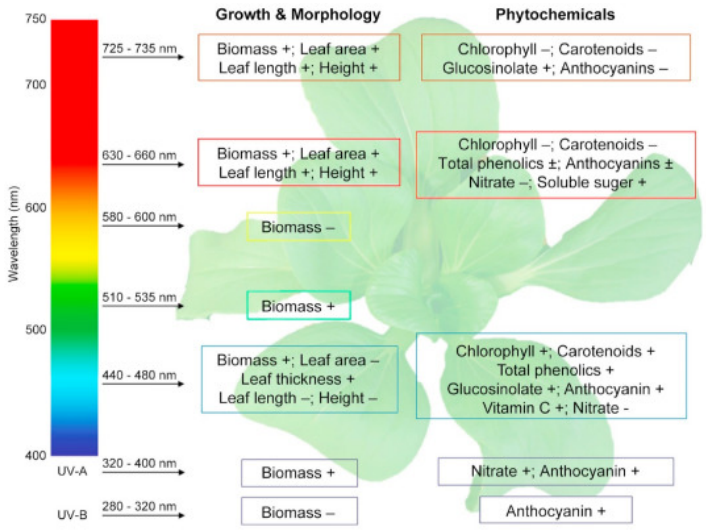


Figura 2.9: Resposta das plantas perante diferentes comprimentos de onda. (Fonte: Wong et al. 2020)

Para cada planta existirá, certamente, um diagrama semelhante ao da figura 2.9 que permitirá extrair conclusões semelhantes. É importante notar que a posição em que a luz se encontra também determina o desenvolvimento mais rápido ou mais lento das plantas.

2.2 Vantagens da agricultura vertical

A agricultura vertical tem benefícios a nível ambiental, económico e social (Benke e Tomkins 2017, Al-Kodmany 2018):

- a nível ambiental:
    1. os alimentos são produzidos sem recurso a pesticidas e herbicidas;
    2. podem ser usadas fontes de energia renováveis, minimizando o uso de combustíveis fósseis, e consequentemente, reduzem-se os níveis de carbono;
    3. uma vez que a produção é feita em ambientes controlados e limitados, há o rejuvenescimento de ecossistemas, e é por isso, sustentável;
    4. as fábricas de plantas podem ser construídas perto de centros urbanos, minimizando a distância de transporte e a pegada de carbono;
- a nível económico:
    1. custos de transporte reduzidos;
    2. custos reduzidos (ou até nulos) em fertilizantes e herbicidas;
    3. a produção não é interrompida por fenómenos meteorológicos extremos (inundações, seca);
    4. a produção pode ser organizada e programada de forma a satisfazer a procura, evitando a sazonalidade de alguns produtos;
- a nível social:
    1. permite empregar em áreas regionais, rurais e remotas;
    2. promove novos empregos em áreas de engenharia, biologia, bioquímica, biotecnologia, construção, investigação e desenvolvimento;
    3. promove um estilo de vida saudável e sustentável: dada a versatilidade das modalidades de produção, é possível construir pequenas estruturas em apartamentos, hóteis, hospitais e escolas, capazes de produzir quantidades suficientes de plantas, minimizando o transporte a centros urbanos para a compra dos mesmos produtos.

Este tipo de agricultura tem ainda benefícios a nível da produção de medicamentos: dada a existência de plantas e ervas com fins medicinais que são encontradas em locais com climas específicos, a agricultura vertical permite simular essas condições e promover o cultivo sustentável de tais plantas a nível local, minimizando a dependência externa.

2.3 Desvantagens da agricultura vertical

As desvantagens deste tipo de agricultura são principalmente a nível económico (Benke e Tomkins 2017, Kozai 2013):

1. a construção de fábricas de plantas em zonas urbanas é mais dispendiosa do que em zonas rurais;
2. os custos de *start-ups* são elevados, assim como todos os equipamentos e tecnologia envolvidos;
3. há perturbação do setor rural (agricultura tradicional).

Capítulo 3

Problemas em agricultura vertical

Neste capítulo, inicialmente faremos uma breve revisão de literatura sobre os modelos de planeamento, e depois vamos expor o problema e introduzir a notação. Aquando da exposição do problema faremos um paralelismo entre o modelo teórico e uma situação realista.

3.1 Revisão bibliográfica

A bibliografia relacionada com modelos de planeamento da produção de plantas em fábricas verticais (VF) é escassa (aliás, o mesmo se passa com os modelos matemáticos de crescimento de plantas, dados relacionados com crescimento de plantas, etc, ...). Destacamos três de múltiplos artigos sobre modelos de planeamento, sendo um deles a base desta dissertação.

Em Glen 1987, é feita uma revisão bibliográfica de modelos matemáticos no planeamento do cultivo de terrenos e de produção pecuária. No entanto, focámo-nos apenas nos modelos de cultivo. Estes modelos são divididos em quatro categorias:

1. modelos de produção de plantas;
2. métodos de planeamento da colheita;
3. métodos para avaliar investimentos;
4. estratégias de controlo de doenças e pragas.

Para cada uma dessas categorias de modelos são abordados vários problemas, nos quais foram usados os métodos - programação linear, programação linear inteira mista, progra-

mação linear inteira, programação estocástica, teoria de portfólios, simulação, processos de Markov, entre outros -, e observações relativas às suas aplicações.

Dada a grande diversidade de problemas presentes no planeamento da produção e a sua elevada complexidade, os modelos e estratégias usados focam-se apenas em aspetos particulares, a sua programação é demorada e difícil, e são poucos os agricultores que são capazes de os usar e interpretar (Glen 1987).

Em Adenso-Díaz e Villa 2021, é feita a modelação matemática do planeamento do cultivo com procura simultânea. Foi proposta uma nova abordagem envolvendo acordos de cooperação a longo prazo, em que os volumes de produto são acordados, e a procura é garantida antecipadamente.

Este problema é semelhante a sistemas de produção de manufaturas, em que o planeamento tem de garantir produção em datas específicas, tornando-se determinístico. Além disso, este modelo tem em consideração a minimização de custos, do risco de não satisfazer a procura, e evitar que haja excedente da produção, uma vez que é perecível.

Com isto, Adenso-Díaz e Villa propuseram um planeamento estratégico de diversificação geográfica, como forma de aumentar a probabilidade de satisfazer a procura, através da maximização da dispersão geográfica da área de cultivo atenuando o risco das condições meteorológicas.

Em Santini et al. 2021 é explorado o *Crop Growth Planning Problem*, abreviado CGPP, que resultou da colaboração com uma empresa de agricultura vertical (VF). No artigo, o CGPP é descrito e trabalhado tendo em consideração vários objetivos.

O trabalho mais próximo é o de Bennell, Martinez e Potts, que não foi publicado em artigo (mas que é importante referir), e que

> Os autores estudam o problema da minimização da procura não satisfeita numa estante, em que cada prateleira está sujeita a diferentes condições de luz e irrigação. Outra decisão a tomar é o meio de crescimento a atribuir a cada prateleira, que influencia tanto a capacidade como as plantas que podem lá crescer. A grande diferença para o CGPP está na presença de uma componente de planeamento no trabalho de Bennell et al. (2017), porque a duração do crescimento da planta pode variar numa dada janela temporal.
>
> (Traduzido de Santini et al. 2021)

É notado que o problema descrito por Bennell, Martinez e Potts 2017 é computacionalmente extenso e complexo, e que os autores recorreram a heurísticas para que fosse possível resolvê-lo.

Existem algumas semelhanças entre o CGPP e problemas de planeamento de máquinas - considerando cada prateleira como máquina, e cada fase de crescimento da planta

como tarefa a ser realizada -, planeamento de máquinas em paralelo (Li e Yang 2009) e com distribuição de tarefas.

No entanto, o CGPP distingue-se dos problemas de planeamento por vários motivos:

1. não é necessário introduzir custos de penalização, uma vez que a procura das plantas determina o tempo em que crescem e estão prontas para a venda;
2. as prateleiras (máquinas) podem ser reconfiguradas, e por isso, podem realizar mais do que uma tarefa;
3. uma só máquina pode ser configurada para satisfazer todas as condições de crescimento da planta;
4. o crescimento das plantas não pode ser suspenso, e por isso, é ininterruptível;
5. o produto é perecível;
6. as máquinas são capacitadas, mas a sua capacidade não é fixa e depende da configuração em que se encontra.

Os autores mostram que o CGPP é um problema *NP-hard*, e concluem que uma abordagem multi-objetivo não é viável por existirem objetivos conflituosos. Deste modo,

> O planeador não pode confiar na hipótese de que otimizando com respeito a um objetivo produzirá soluções aceitáveis com respeito a outro objetivo.
>
> (Traduzido de Santini et al. 2021)

Por fim, adotamos a notação e descrição dos problemas em Santini et al. 2021.

3.2 Estrutura e organização de uma fábrica vertical

Queremos operar uma fábrica de plantas que pretende fazer crescer uma ou várias colheitas para satisfazer a procura. Para cada planta da colheita é necessário saber as condições mais favoráveis de crescimento - temperatura, luz, humidade, solução de sais, ar, entre outras. As plantas crescem em prateleiras empilhadas, que são colocadas em estantes. Cada prateleira tem uma configuração única de fatores de crescimento e pode ser alterada, manual ou automaticamente.

Os problemas que vão ser abordados, descritos na Secção 3.2.2 após a introdução das variáveis de decisão e das restrições, referem-se a métodos de planeamento da colheita e são os seguintes:

1. minimização do número de movimentos;
2. minimização do número de reconfigurações;
3. minimização da procura não satisfeita;
4. minimização do número de prateleiras usadas.

3.2.1 Variáveis de decisão

Seja C o conjunto das plantas de uma certa colheita, S o conjunto das prateleiras, e D o horizonte temporal da procura, $D = \{1, \ldots, \bar{d} - 1\}$. Cada $d \in D$ representa um dia, $\bar{d}$ denota o último dia possível para a entrega de plantas (portanto, no dia $\bar{d}+1$, a fábrica está encerrada); logo, $\bar{d} - 1$ indica o último dia em que as plantas podem crescer; $D' = \{0\} \cup D$ representa o horizonte estendido de crescimento, onde o dia 0 indica o dia em que as plantas são semeadas. Vamos assumir que, para cada $d \in D \cup \{\bar{d}\}$ temos procura de $p_{c,d}$ unidades de $c \in C$ (naturalmente, $p_{c,d} \in \mathbb{N}_0$).

Designamos por σ o edifício (banco) onde estão guardadas as sementes, por τ o edifício ou armazém onde a produção é guardada após o crescimento, e assumimos que ambos constituem prateleiras. Geralmente, a sementeira desenvolve-se num espaço físico (unidade específica) diferente do das restantes fases de crescimento (Shamshiri et al. 2018). Para os modelos que aqui se vão apresentar, este facto é irrelevante porque não interessa o local físico onde cada fase de crescimento ocorre, mas sim os movimentos das paletes com as plantas entre os diferentes espaços (que podem ser ou não comuns) associados ao crescimento. $S' = \{\sigma, \tau\} \cup S$ denotará o conjunto de todas as prateleiras, onde σ e τ são entendidos como prateleiras.

Vamos supor $C = \{\text{morango}, \text{alface}, \text{pimento}\}$. Queremos fazer crescer cada uma das plantas para satisfazer a procura no período de $\bar{d} = 100$ dias.

Atendendo às características das prateleiras (dimensões), iremos definir o conjunto das prateleiras em que a planta c pode crescer, $S_c = \{s \in S : \delta_{c,s} = 1\}$, onde $\delta_{c,s} \in \{0, 1\}$ é um parâmetro de compatibilidade entre a planta e a prateleira dado por,

$$\delta_{c,s} = \begin{cases} 1, & \text{se a planta } \textit{c} \text{ pode crescer na prateleira } \textit{s}, \\ 0, & \text{caso contrário.} \end{cases}$$

Assim sendo, consideremos que temos 30 prateleiras. Podemos identificar quatro fases de crescimento para o morango e para o pimento, enquanto que na alface temos três.

Figura 3.1: Fases de crescimento do morango (da esquerda para a direita: (a) germinação, (b) floração, (c) crescimento do fruto, (d) maturação). (Fonte: *Pixabay.com*)

Figura 3.2: Fases de crescimento da alface (da esquerda para a direita: (a) germinação, (b) crescimento das primeiras folhas, (c) maturação). (Fonte: *Pixabay.com*)

Figura 3.3: Fases de crescimento do pimento (da esquerda para a direita: (a) germinação, (b) floração, (c) crescimento do pimento, (d) maturação). (Fonte: *Pixabay.com*)

Cada planta de uma colheita tem diferentes estados de crescimento que requerem condições específicas[1]. Assim sendo, vamos denotar por γ_c o número de dias de crescimento para cada planta, e $k_{c,g} \in K$ a configuração requerida para a planta c no seu g-ésimo dia de crescimento ($g \in \{1, \ldots, \gamma_c\}$). K designa o conjunto de todas as configurações possíveis dos diferentes fatores que intervêm no crescimento das plantas. As configurações das prateleiras e o estado de crescimento da planta afetam a capacidade das prateleiras. Assim, $q_{s,k} \in \mathbb{N}$ indica a capacidade da prateleira s (número de paletes com plantas) com a configuração k, sendo uma limitação da fábrica.

Para o nosso caso prático, consideremos que $K = \{k_1, k_2, k_3, k_4\}$ e, que o morango demora 30 dias a crescer ($\gamma_{\text{morango}} = 30$), a alface demora 20 dias a crescer ($\gamma_{\text{alface}} = 20$),

[1] Estas condições podem ser determinadas através de modelos matemáticos (Bessonov, Volpert e Volpert 2006; Roose 2008), mas que não são objeto de estudo nesta dissertação.

e o pimento demora 40 dias a crescer ($\gamma_{\text{pimento}} = 40$)[2]. Além disso, cada prateleira permite crescer 20 unidades ($q_{s,\,k} = 20$).

Na realidade, os trabalhadores evitam mover as plantas, pois a movimentação pode danificar e interferir no crescimento.

Para simplificar, vamos assumir que as prateleiras são capazes de suportar as plantas durante todo o ciclo de crescimento, e que existem vários tipos de prateleiras (altas, médias, baixas). Deste modo, passamos a ter o conjunto dos tipos de prateleiras, T, em vez do conjunto S.

Analogamente ao conjunto S, definimos:

- parâmetro de compatibilidade entre tipo de prateleira t e planta, $\delta_{c,t}$, como

$$\delta_{c,t} = \begin{cases} 1, & \text{se a planta } c \text{ pode crescer na prateleira de tipo } t, \\ 0, & \text{caso contrário.} \end{cases}$$

- o conjunto dos tipos de prateleiras em que a planta pode crescer, $T_c = \{t \in T : \delta_{c,t} = 1\}$;
- $T' = \{\sigma, \tau\} \cup T$ denotará o conjunto de todos os tipos de prateleiras.

O número de paletes que cada prateleira suporta, depende do tipo e da configuração a que ela se encontra.

Seja $S_t \subseteq S$ o conjunto das prateleiras s do tipo t. Na figura 3.4, temos $S_{t_1} = \{s_1\,, s_5\,, s_9\}$. Seja $q_{t,\,s,\,k}$ o número (máximo) de paletes que a prateleira $s \in S$ de tipo $t \in T$ admite, quando está a funcionar na condição $k \in K$. Por exemplo, na figura 3.4, tem-se $q_{t_1,\,s_1,\,k_3} = 4$.

Para simplificar o modelo e porque, na prática, as fábricas adquirem prateleiras do mesmo tipo idênticas, iremos considerar que o número de paletes que cada prateleira leva depende somente do seu tipo e da configuração a que ela se encontra. Seja, então, $q_{t,\,k}$ o número de paletes que uma prateleira de tipo t leva na configuração k.

Denotamos por n_t o número de prateleiras do tipo t, sendo, portanto, um dado conhecido do problema.

[2] O tempo de crescimento das plantas numa fábrica vertical é inferior ao tempo que demoram a crescer na agricultura tradicional. Sendo assim, para cada planta, tomamos o tempo mínimo de crescimento.

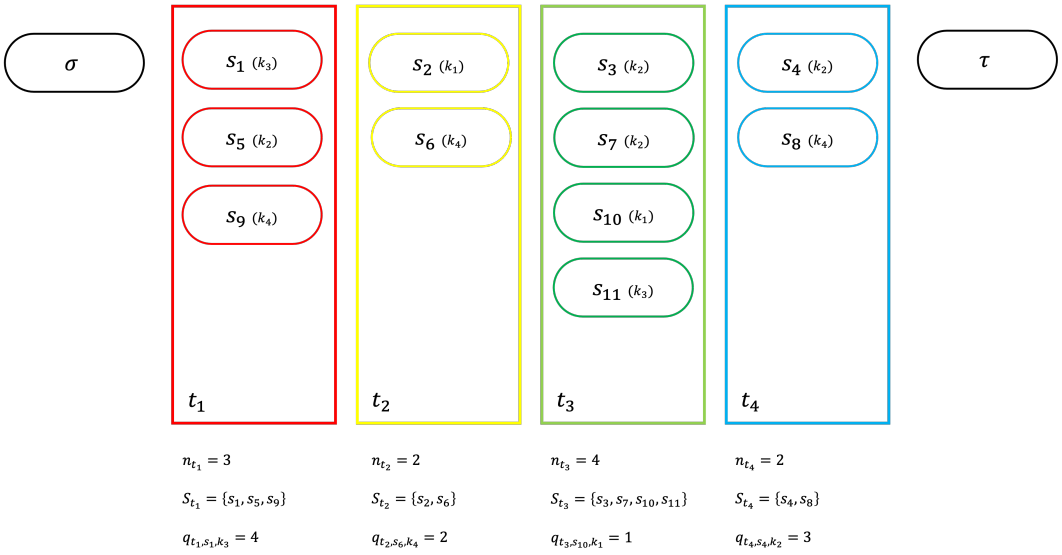


Figura 3.4: Ilustração de uma fábrica com onze prateleiras $(s_1, s_2, \ldots, s_{11})$, cada uma delas com uma configuração (k_i), com $k_i \in K = \{k_1, k_2, k_3, K_4\}$ distribuídas por quatro tipos de prateleiras $(T = \{t_1, t_2, t_3, t_4\})$.

Usaremos dois conjuntos de variáveis de decisão:

1. $x^{c,g}_{t_1,d,t_2} \in \mathbb{N}_0$: representa o número de paletes da planta $c \in C$ que estão no g-ésimo dia de crescimento, em prateleiras de tipo t_1 no dia d e que irão para prateleiras do tipo t_2 no dia $d+1$;

2. $y_{t,d,k} \in \mathbb{N}_0$: representa o número de prateleiras do tipo t, na configuração k no dia d.

Suponhamos que queremos fazer crescer alface para satisfazer uma encomenda. A alface tem três fases de crescimento que requerem diferentes configurações e diferentes tempos de crescimento. Um exemplo de esquema de uma fábrica com essas condições é dado na figura 3.5.

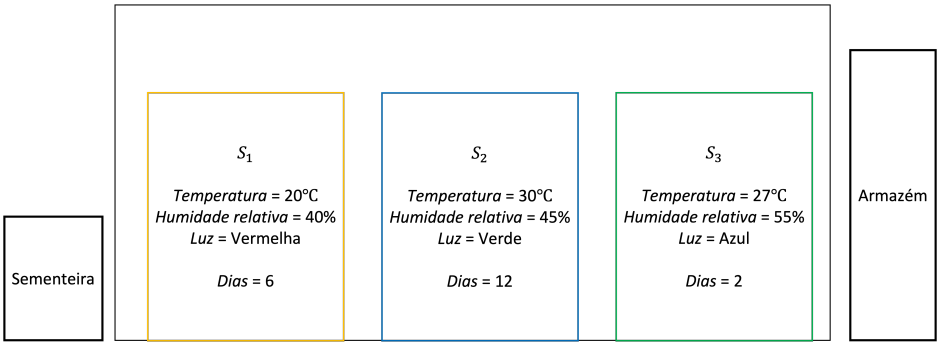


Figura 3.5: Exemplo de esquema de uma fábrica de plantas, tempo total de crescimento de 20 dias e três configurações distintas (distinguidas pela cor).

3.2.2 Funções objetivo

Vamos indicar as funções objetivo que podem ser consideradas.

3.2.2.1 Minimização do número de movimentos (MinM)

A movimentação de plantas entre prateleiras é demorada e pode danificá-las. De forma a evitar que tal ocorra, prefere-se que as plantas permaneçam o máximo de tempo possível na mesma prateleira, alterando-se apenas as configurações conforme o crescimento da planta.

Se usarmos as variáveis indexadas com o tipo de prateleiras (T) será bastante difícil modelar os movimentos das prateleiras. Assim, tomaremos as variáveis indexadas pelas prateleiras (S). Deste modo, a função objetivo é

$$\min \quad \sum_{c\in C} \sum_{g=1}^{\gamma_c} \sum_{\substack{s_1,s_2\in S_c \\ s_1\neq s_2}} \sum_{d\in D} x^{c,g}_{s_1,d,s_2} \,. \tag{3.1}$$

Observação. Podíamos reescrever a função objetivo anterior (3.1) usando o conjunto T, notando que:

$$x^{c,g}_{t_1,d,t_2} = \sum_{s_1\in S_{t_1}} \sum_{s_2\in S_{t_2}} x^{c,g}_{s_1,d,s_2} \,. \tag{3.2}$$

3.2.2.2 Minimização do número de reconfigurações (MinR)

Mudar as configurações das prateleiras é demorado e está sujeito a erros. Contrariamente ao caso anterior, prefere-se que as prateleiras tenham configurações estáveis, mudando as plantas por forma a estarem na prateleira com as condições ideais para a fase de crescimento.

Deste modo, a função objetivo que reflete essa necessidade é

$$\min \quad \sum_{t\in T} \sum_{d=1}^{\bar{d}-2} \frac{1}{2} \sum_{k\in K} \left| y_{t,d,k} - y_{t,d+1,k} \right| , \tag{3.3}$$

onde,

- o fator $\frac{1}{2}$ surge da mudança de configuração de uma prateleira mudar o valor de duas variáveis y de uma vez só;

- $|y_{t,d,k} - y_{t,d+1,k}|$: como não existe uma restrição que determine a configuração de uma prateleira não usada, a função objetivo fará com que elas fiquem com a última configuração usada.

Assim, a função objetivo tomará a forma

$$\min \quad \sum_{t \in T} \sum_{d=1}^{\bar{d}-2} \frac{1}{2} \sum_{k \in K} w_{t,d,k}\,, \tag{3.4}$$

onde $w_{t,d,k} = |y_{t,d,k} - y_{t,d+1,k}|$ (ou $|y_{t,d+1,k} - y_{t,d,k}|$) $\in \mathbb{N}$.

3.2.2.3 Minimização da procura não satisfeita (MinUD)

Se não for possível garantir que haja um planeamento admissível para a procura, podemos decidir não satisfazer alguma parte dela.

Neste caso, introduz-se uma nova variável $u_{c,d} \in \mathbb{N}$, que indica a quantidade de procura não satisfeita da planta c no dia $d \in D \cup \{\bar{d}\}$. Teremos de mudar as restrições (3.10) e (3.12), de modo a contemplarem as alterações:

- a restrição 3.10 fica

$$\sum_{t \in T_c} x^{c,\gamma_c}_{t,d-1,\tau} + u_{c,d} = p_{c,d} \quad \forall c \in C, \forall d \in D \cup \{\bar{d}\}; \tag{3.5}$$

- a restrição 3.12 fica

$$\sum_{t \in T_c} x^{c,0}_{t,d-\gamma_c-1,t} + u_{c,d} = p_{c,d} \quad \forall c \in C, \forall d \in \{\gamma_c + 1, \ldots, \bar{d}\}. \tag{3.6}$$

Definindo $\omega_{c,d} \in \mathbb{R}^+$, como o custo de perda de uma unidade de procura da planta c no dia d, a função objetivo é a seguinte:

$$\min \quad \sum_{c \in C} \sum_{d=1}^{\bar{d}} \omega_{c,d}\, u_{c,d}. \tag{3.7}$$

3.2.2.4 Minimização do número de prateleiras usadas (MinS)

De um ponto de vista estratégico, poderemos querer operar uma fábrica de plantas usando o menor número de prateleiras possível e que nos permita satisfazer a procura.

Assim, introduzimos uma configuração fictícia, $k_0 \in K$ que representa uma prateleira não usada, e um novo conjunto de variáveis $v_t \in \mathbb{N}$ que indicam o número de prateleiras do tipo $t \in T$ usadas na solução. Com isto, as capacidades das prateleiras associadas a k_0 são 0, $q_{t,k_0} = 0, \forall t \in T$.

A função objetivo é

$$\min \quad \sum_{t \in T} v_t \,, \tag{3.8}$$

adicionando a restrição

$$v_t \geq \sum_{k \in K \setminus \{k_0\}} y_{t,d,k} \,, \quad \forall t \in T, \forall d \in D \tag{3.9}$$

para garantir que que as variáveis v_t tomam os valores corretos.

3.2.3 Restrições

As restrições impostas são transversais a todos os problemas indicados.

3.2.3.1 Satisfação da procura

Procuramos garantir que existam e já estejam armazenadas unidades da planta $c \in C$ para satisfazer a procura no dia d:

$$\sum_{t \in T_c} x^{c,\gamma_c}_{t,d-1,\tau} = p_{c,d}, \quad \forall c \in C, \forall d \in D \cup \{\bar{d}\}. \tag{3.10}$$

3.2.3.2 Restrição de capacidade

Pretendemos que num determinado dia, haja uma quantidade admissível de plantas em crescimento (a capacidade de uma prateleira não é fixa, mas depende da configuração usada):

$$\sum_{t_2 \in T'} \sum_{\substack{c \in C \\ \delta_{c,\,t_1}=1 \\ \delta_{c,\,t_2}=1}} \sum_{\substack{g \in \{1,\ldots,\gamma_c\} \\ k = k_{c,\,g}}} x^{c,g}_{t_1,d,t_2} \leq q_{t_1,\,k} \cdot y_{t_1,\,d,\,k} \quad \forall t_1 \in T, \forall d \in D, \forall k \in K. \tag{3.11}$$

3.2.3.3 Restrição de plantação

A procura determina quando deve ser feita a colheita, e por isso, determina indiretamente, quando deve ser feita a plantação:

$$\sum_{t \in T_c} x^{c,0}_{\sigma,d-\gamma_c-1,t} = p_{c,d}\,, \quad \forall c \in C,\, \forall d \in \left\{\gamma_c + 1, \ldots, \bar{d}\right\}. \tag{3.12}$$

3.2.3.4 Restrição de equilíbrio do fluxo

Queremos garantir que qualquer quantidade de unidades de plantas que cresçam em prateleiras de um tipo num determinado dia seja enviada para a mesma prateleira ou outras prateleiras no dia seguinte ($T'_c = \{\sigma, \tau\} \cup T_c$):

$$\sum_{t_2 \in T'_c} x^{c,g}_{t_1,d-1,t_2} = \sum_{t_2 \in T'_c} x^{c,g+1}_{t_1,d,t_2}, \quad \forall c \in C,\, \forall g \in \{0, \ldots, \gamma_c\},\, \forall t_1 \in T_c,\, \forall d \in D. \tag{3.13}$$

Esta restrição garante que o dia de crescimento aumenta em 1, cada vez que o dia passa.

3.2.3.5 Restrição de configuração

Com esta condição, garantimos que não temos mais configurações que prateleiras, para cada tipo em cada dia:

$$\sum_{k \in K} y_{t,\,d,\,k} \leq n_t, \quad \forall t \in T,\, \forall d \in D. \tag{3.14}$$

3.2.4 Fixação de variáveis

Variáveis do tipo $x^{c,g}_{t_1,d,t_2}$

Ao fixar algumas variáveis $x^{c,g}_{t_1,d,t_2}$ permitiremos que algumas delas possam ser ignoradas, pelo facto de não determinarem condições viáveis, ou porque não podem tomar qualquer outro valor para além de 0 na solução otimal. Deste modo, tomaremos como 0:

1. todas as variáveis que tenham como destino a sementeira (σ), ou que tenham como origem o armazém (τ), ou ainda, que tenham início na sementeira, e que após d dias, tenham como destino o armazém:

$$\begin{cases} x^{c,g}_{t,d,\sigma} = x^{c,g}_{\tau,d,t} = 0 & \forall d \in D', \forall c \in C, \forall t \in T_c \cup \{\sigma, \tau\}, \forall g \in \{0, \ldots, \gamma_c\} \\ x^{c,g}_{\sigma,d,\tau} = 0 & \forall d \in D', \forall c \in C, \forall g \in \{0, \ldots, \gamma_c\} \end{cases} \quad \text{(FV1)}$$

2. todas as variáveis procedentes de σ correspondentes a dias de crescimento diferentes de 0, ou todas as variáveis que tenham como destino τ e que ainda não estejam amadurecidas:

$$\begin{cases} x^{c,g}_{\sigma,d,t} = 0 & \forall d \in D', \forall c \in C, \forall t \in T_c \cup \{\sigma, \tau\}, \forall g \in \{1, \ldots, \gamma_c\} \\ x^{c,g}_{t,d,\tau} = 0 & \forall d \in D', \forall c \in C, \forall t \in T_c \cup \{\sigma, \tau\}, \forall g \in \{0, \ldots, \gamma_c - 1\} \\ x^{c,g}_{t_1,d,t_2} = 0 & \forall d \in D, \forall c \in C, \forall t_1,\, t_2 \in T_c, \forall g \in \{0,\, \gamma_c\} \end{cases} \quad \text{(FV2)}$$

3. todas as variáveis com 0 dias de crescimento devem sair de σ, e todas as variáveis que correspondentes a culturas amadurecidas devem de ir para τ:

$$\begin{cases} x^{c,0}_{t_1,d,t_2} = 0 & \forall t_1 \in T' \setminus \{\sigma\}, \forall d \in D', \forall t_2 \in T', \forall c \in C \\ x^{c,\gamma_c}_{t_1,d,t_2} = 0 & \forall t_1 \in T', \forall d \in D', \forall t_2 \in T' \setminus \{\tau\}, \forall c \in C \end{cases} \quad \text{(FV3)}$$

4. todas as variáveis em que as plantas não conseguiram amadurecer a tempo da colheita; chamando $d'_c = \max_{d \in D}\{d :\, p_{c,d} > 0\}$ ao último dia com alguma procura pela planta $c \in C$:

$$x^{c,g}_{t_1,d,t_2} = 0 \qquad \forall c \in C, \forall t_1, t_2 \in T_c \cup \{\sigma, \tau\}, \forall d \in \{d'_c - \gamma_c + 1, \ldots, d'_c\}, \quad \forall g \in \{0, \ldots,\, \gamma_c - (d'_c - d)\} \quad \text{(FV4)}$$

5. todas as variáveis correspondentes a dias de crescimento não admissíveis:

$$x^{c,g}_{t_1,d,t_2} = 0 \quad \forall c \in C, \forall t_1, t_2 \in T_c \cup \{\sigma, \tau\}, \forall d \in D' : d < \gamma_c, \forall g \in \{d+1, \ldots, \gamma_c\}. \quad \text{(FV5)}$$

Como a procura limita as unidades de plantas que são plantadas e colhidas, podemos ainda limitar a quantidade de plantas que sai de σ e que é armazenada (τ):

$$\begin{cases} x^{c,0}_{\sigma,d-\gamma_c-1,t} & \leq p_{c,d} \qquad \forall c \in C,\, \forall t \in T_c,\, \forall d \in \{\gamma_c + 1, \ldots, \bar{d}\} \\ x^{c,\gamma_c}_{t,d-1,\tau} & \leq p_{c,d} \qquad \forall c \in C,\, \forall t \in T_c,\, \forall d \in D \cup \{\bar{d}\}. \end{cases} \tag{FV6}$$

Variáveis do tipo $y_{t,d,k}$

Como são conhecidas as fases de crescimento e a procura de cada planta, sabemos quantas unidades de plantas requerem uma dada configuração para um dado dia. Designando por $\eta_{d,\,k} \in \mathbb{N}$, o número de unidades de cada planta, que requerem a configuração k no dia d, temos

$$\eta_{d,\,k} = \sum_{d'=d+1}^{d+\gamma_c-1} \sum_{\substack{c \in C: \\ k=k_{c,\,\gamma_c-(d'-d-1)}}} p_{c,d} \quad \forall d \in D,\, \forall k \in K.$$

Assim, podemos fixar todas as variáveis y que correspondem a configurações não necessárias:

$$y_{t,d,k} = 0 \quad \forall t \in T,\, \forall d \in D,\, \forall k \in K : \eta_{d,\,k} = 0. \tag{FV7}$$

3.3 Um caso simples

De forma a compreender melhor este modelo – variáveis de decisão, restrições, incompatibilidades, etc. – vamos imaginar uma fábrica de plantas com um número reduzido de prateleiras a operar num período de poucos dias.

Suponhamos uma fábrica que labora/trabalha num período de 10 dias ($\bar{d} = 10$). Esta fábrica produz, por exemplo, cebolinho[3] que demora 6 dias a ser produzido ($\gamma = 6$).

Vamos supor que a fábrica dispõe somente de 4 tipos de prateleiras distintas, a saber, t_1, t_2, t_3 e t_4, onde se colocam as paletes de cebolinho ao longo da sua fase de produção (por exemplo, prateleiras de dimensões pequenas, médias ou altas). Seja T o conjunto dos tipos de prateleiras, isto é, $T = \{t_1\,, t_2\,, t_3\,, t_4\}$.

Além disso, podemos ter mais do que uma prateleira de cada tipo $t \in T$. Assim, n_t denota o número de prateleiras de tipo t. No nosso caso, vamos assumir $n_t = 1\,,\ \forall t \in T$.

[3] Erva aromática caracterizada pelas folhas muito finas, altas, verdes e cilíndricas, com crescimento vertical

Na realidade, pode existir mais do que uma prateleira de cada tipo, por exemplo, $n_{t_1} = 2$, ou seja, há duas prateleiras de tipo t_1. Nesse caso, define-se S_t como o conjunto de prateleiras do tipo t.

As prateleiras funcionam sob configurações/condições específicas e distintas de luz, água, sais minerais, temperatura, etc.. Vamos assumir que as prateleiras dispõe de 4 destas configurações possíveis: I_1, I_2, I_3, e I_4. Seja K o conjunto das configurações possíveis, isto é, $K = \{I_1, I_2, I_3, I_4\}$.

Neste caso, iremos ainda assumir que $q_{t,k} = 5$, $\forall t \in T \ \forall k \in K$, isto é, a prateleira de tipo t na configuração k admite, no máximo, 5 paletes. A figura 3.6 ilustra 4 paletes numa prateleira, por exemplo, do tipo t_2 na configuração I_3.

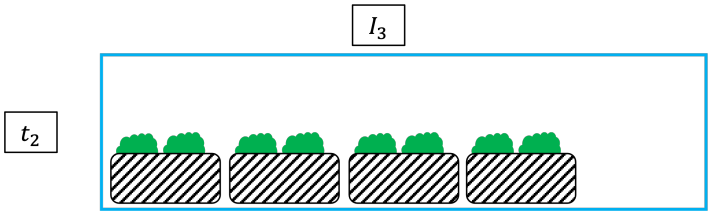


Figura 3.6: Ilustração, por exemplo, de uma prateleira do tipo t_2 , na configuração I_3 , com 4 paletes, onde $q_{t_2,\,I_3} = 5$ paletes.

Vamos supor ainda que a produção ocorre ao longo de quatro fases: G: germinação; C1: crescimento 1; C2: crescimento 2; E: maturação (tratamento ou embelezamento final).

Neste modelo simplificado, vamos assumir que cada prateleira está numa só configuração, assim definida: a prateleira de tipo t_k funciona na condição I_k $(k = 1, 2, 3, 4)$.

A produção da fábrica está organizada de forma a que as encomendas são expressas pelo número de paletes de pés de cebolinho que o cliente pretende. Todas as paletes contêm o mesmo número de pés de cebolinho (por exemplo, 25).

A Tabela 3.1 sumariza todas estas condições.

Fase de crescimento	Duração (dias)	Prateleira	Configuração
Germinação	1	t_1	I_1
Crescimento 1	2	t_2	I_2
Crescimento 2	2	t_3	I_3
Maturação	1	t_4	I_4

Tabela 3.1: Duração, localização nas prateleiras e configuração de cada fase de crescimento.

Assim sendo, como o cebolinho demora seis dias a crescer, a procura só pode ocorrer

a partir do sétimo dia. Ou seja, a procura diária é dada pelo vetor linha:

$$P = [p_1\,, p_2\,, p_3\,, p_4\,, p_5\,, p_6\,, p_7\,, p_8\,, p_9\,, p_{10}] = [0, 0, 0, 0, 0, 0, 1, 0, 0, 0].$$

Neste sentido, a produção de cebolinho segue uma sucessão de configurações, que se podem repetir ou não. Para esta entrega, como o cebolinho precisa de 6 dias para o crescimento, vão percorrer a sequência $I_1 - I_2 - I_2 - I_3 - I_3 - I_4$ de configurações. Assim sendo, $K_g \equiv \{I_1,\ I_2,\ I_2,\ I_3,\ I_3,\ I_4\}$ (ver Tabela 3.2).

Dias de laboração	0	1	2	3	4	5	6	7	8	9	$\bar{d} = 10$
Procura		0	0	0	0	0	0	1	0	0	0
Horizonte de crescimento		G	C1	C1	C2	C2	E				
Tipo de prateleira	σ	t_1	t_2	t_2	t_3	t_3	t_4	τ			
Configuração		I_1	I_2	I_2	I_3	I_3	I_4				

Tabela 3.2: G: germinação; C1: crescimento 1; C2: crescimento 2; E: embelezamento; σ : prateleiras das sementes; τ : prateleiras onde a produção é guardada após o crescimento.

As variáveis de decisão deste problema dependem do dia d da procura diária, do dia g de crescimento, das prateleiras r e s em que se encontram os pés de cebolinho, e da configuração k das prateleiras. As variáveis de decisão são distinguidas em dois tipos:

1. variáveis que determinam o movimento das paletes entre as prateleiras dos vários tipos, $x^g_{r,d,s}$;

2. variáveis associadas ao número de prateleiras de tipo t usadas para cada configuração, $y_{t,d,k}$.

As variáveis de decisão $x^g_{r,d,s}$ representam o número de paletes com plantas (portanto, $x^g_{r,d,s} \in \mathbb{N}_0$) no dia d que estão no g-ésimo dia de crescimento na prateleira de tipo $r \in \{t_1, t_2, t_3, t_4\}$ e que, no dia $d + 1$, vão para a prateleira de tipo $s \in \{t_1, t_2, t_3, t_4\}$ (ver figura 3.7).

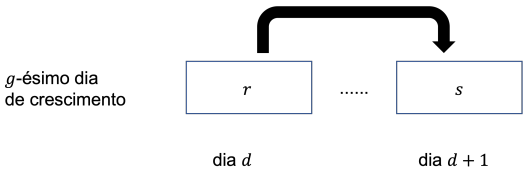


Figura 3.7: Variável de decisão $x^g_{r,d,s}$.

Suponhamos que $x^3_{t_2,3,t_3} = 1$, isto é, existe uma palete de cebolinho que está no seu terceiro dia de crescimento (na fase C1) na prateleira de tipo t_2, e que no dia quatro vai ser movida para a prateleira de tipo t_3.

As variáveis de decisão $y_{t,d,k}$ indicam o número de prateleiras de tipo $t \in T$, que no dia d, encontra-se na configuração $k \in K \equiv \{I_1, I_2, I_3, I_4\}$. Por sua vez, assumimos que cada tipo de prateleira só admite uma e uma só condição. Vamos supor que $y_{t_3,4,I_1} = 0$, $y_{t_3,4,I_2} = 0$, $y_{t_3,4,I_3} = 1$, $y_{t_3,4,I_4} = 0$. Isto significa que no dia 4, há uma prateleira de tipo t_3 na condição I_3 e nenhuma prateleira de tipo t_3 nas condições I_1 , I_2 , I_4.

Além disso, vamos assumir que os movimentos de paletes entre prateleiras são efetuados de tal forma que:

1. todas as prateleiras $t \in T$ podem receber de σ e enviar para τ, mas os movimentos recíprocos correspondentes não são possíveis;

2. o movimento das paletes segue a ordem natural do crescimento das plantas (isto é, a ordem imposta pelas sucessivas fases desde a germinação até à etapa final do crescimento), isto é, como as configurações $k \in K$ são distintas, o movimento oposto ao do crescimento das plantas não é permitido (ver figura 3.8);

3. as plantas que requerem vários dias na mesma configuração não efetuam qualquer movimento entre prateleiras (representado pelo *loop* na figura 3.8).

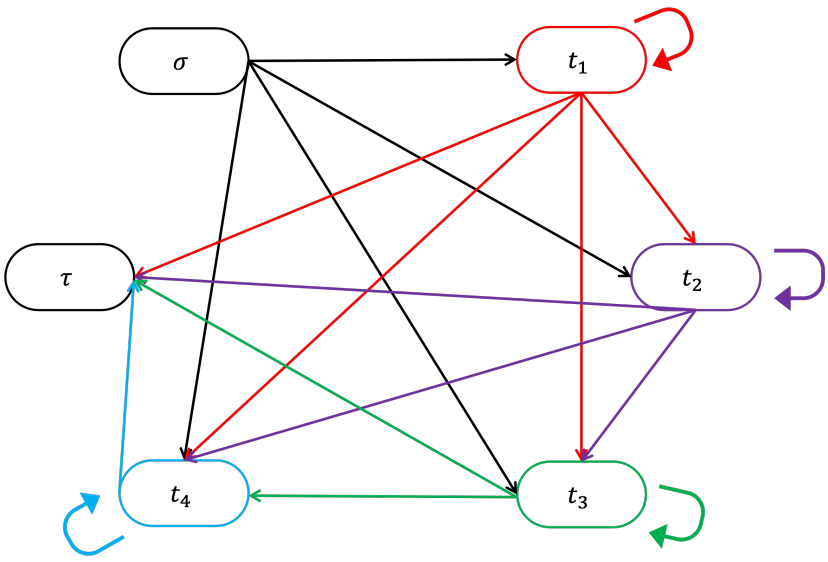


Figura 3.8: Esquema do caso simples para quatro prateleiras.

3.3.1 Minimização do número de movimentos de paletes

Tendo definidas as variáveis de decisão, pretendemos, agora, minimizar o número de movimentos de paletes entre as prateleiras dos vários tipos, isto é, queremos:

$$\min \quad \sum_{g=1}^{6} \sum_{\substack{r,s \in T \\ r \neq s}} \sum_{d \in \{1,2,\ldots,9\}} x^{g}_{r,d,s} \,. \tag{3.15}$$

Notar que temos $6 \times \left(\frac{4^2-4}{2}\right) \times 9 = 324$ variáveis de decisão que entram na função objetivo. O número total de variáveis ($x^{g}_{r,d,s}$ e $y_{t,d,k}$) alocadas no computador é de:

$$\left[7 \times \left(\frac{6^2-6}{2}+6\right) \times 10\right] + [4 \times (10-1) \times 4] = 1614.$$

A solução trivial deste problema é apresentada de seguida, notando que são somente indicadas as variáveis de decisão que tomam valores não nulos:

$$x^{0}_{\sigma,0,t_1} = x^{1}_{t_1,1,t_2} = x^{2}_{t_2,2,t_2} = x^{3}_{t_2,3,t_3} = x^{4}_{t_3,4,t_3} = x^{5}_{t_3,5,t_4} = x^{6}_{t_4,6,\tau} = 1,$$
$$y_{t_1,1,I_1} = y_{t_2,2,I_2} = y_{t_2,3,I_2} = y_{t_3,4,I_3} = y_{t_3,5,I_3} = y_{t_4,6,I_4} = 1.$$

De seguida, apresentamos as restrições associadas a este problema. As condições fronteira envolvem sete condições de fixação de variáveis que, mais abaixo estão referenciadas por FV1, FV2, ..., FV7. Para ganharmos uma maior familiaridade com os diversos índices das variáveis de decisão, no Apêndice A, apresentamos as várias restrições associadas aos diferentes índices. No Apêndice B.1, apresentamos o código *Python* desenvolvido.

Procura:

$$\sum_{s \in T} x^{6}_{s,d-1,\tau} = p_d \quad \forall d \in D \cup \{\bar{d}\} = \{1,\ldots,10\}$$

Capacidade:

$$\sum_{r \in T'} \sum_{\substack{g \in \{1,\ldots,6\} \\ k_g = k}} x^{g}_{s,d,r} \leq 5 \cdot y_{s,\,d,\,k} \quad \forall s \in T\,, \forall k \in K\,, \forall d \in \{1,2,\ldots,9\}$$

Plantação:

$$\sum_{s \in T} x^{0}_{\sigma,d-6-1,s} = p_d \quad \forall d \in \{7,\ldots,10\}$$

Configuração:

$$\sum_{k\in K} y_{s,d,k} \leq 1 \quad \forall s \in T\,, \forall d \in D$$

Equilíbrio:

$$\sum_{r\in T'} x^{g}_{r,d-1,s} = \sum_{r\in T'} x^{g+1}_{s,d,r} \quad \forall g \in \{0,\ldots,6\}\,, \forall s \in T\,, \forall d \in D$$

Fixação de variáveis:

FV1: $x^{g}_{t,d,\sigma} = x^{g}_{\tau,d,t} = x^{g}_{\sigma,d,\tau} = 0, \quad \forall t \in T'$, $\forall g \in \{0,\ldots,6\}$, $\forall d \in \{0,1,\ldots,9\}$.

FV2:

$$\begin{aligned} x^{g}_{\sigma,d,t} &= 0, \qquad \forall t \in T',\ \forall g \in \{1,\ldots,6\},\ \forall d \in \{1,\ldots,9\} \\ x^{g}_{t,d,\tau} &= 0, \qquad \forall t \in T',\ \forall g \in \{0,\ldots,5\},\ \forall d \in \{0,1,\ldots,9\} \\ x^{g}_{t_1,d,t_2} &= 0, \qquad \forall t_1,\ t_2 \in T,\ \forall g \in \{0,\ 6\},\ \forall d \in \{1,\ldots,9\} \end{aligned}$$

FV3: $\forall d \in \{0,1,\ldots,9\}$

$$\begin{aligned} x^{0}_{t_1,d,t_2} &= 0 \qquad \forall\, t_1 \in T' \setminus \{\sigma\},\ \forall\, t_2 \in T' \\ x^{6}_{t_1,d,t_2} &= 0 \qquad \forall\, t_1 \in T',\ \forall\, t_2 \in T' \setminus \{\tau\} \end{aligned}$$

FV4: $d' = \max_{d\in\{1,\ldots,9\}}\{d :\ p_d > 0\}$ o último dia com procura. Temos $d' = 7$.

$$\begin{aligned} &x^{g}_{s,d,r} = 0 \quad \forall s,r \in T,\ \forall d \in \{d'-6+1,\ldots,d'\}, \forall g \in \{0,\ldots,\ 6-(d'-d)\},\ \text{ou seja,} \\ &x^{g}_{s,d,r} = 0 \quad \forall r,s \in T,\ \forall d \in \{2,3,4,5,6,7\}, \forall g \in \{0,\ldots,\ d-1\} \end{aligned}$$

FV5: $x^{g}_{r,d,s} = 0 \quad \forall r,s \in T'$, $\forall d \in D' : d < 6$, $\forall g \in \{d+1,\ldots,6\}$.

FV6:

- $x^{0}_{\sigma,d-6-1,r} \leq p_d \quad \forall r \in T$, $\forall d \in \{6+1,\ldots,10\}$.
- $x^{6}_{s,d-1,\tau} \leq p_d \quad \forall s \in T$, $\forall d \in \{1,2,\ldots,10\}$.

FV7: $\eta_{d,\,k} \in \mathbb{N}$, o número de paletes que requerem a configuração k no dia d, temos:

$$\eta_{d,\,k} = \sum_{d'=d+1}^{d+6} \sum_{k_{6-(d'-d-1)}=k} p_{d'} \quad \forall d \in \{1,\ldots,9\},\ \forall k \in K.$$

Então, $\quad y_{s,d,k} = 0 \quad \forall s \in T$, $\forall d \in D$, $\forall k \in K : \eta_{d,\,k} = 0$.

Capítulo 4

Simulação Computacional e Análise dos Resultados

Neste capítulo, vamos modelar dois casos associados ao problema MinM, faremos algumas considerações iniciais, introduziremos as variáveis, restrições e realizaremos simulações. Para as simulações, escolhemos tratar apenas de um problema abordado no capítulo anterior, o problema MinM – minimização do número de movimentos.

Os casos descritos são implementados no *Python*, usando o pacote *Pyomo - Python Optimization Modeling Objects* (Hart et al. 2021). Tratando-se de um problema de programação inteira mista, usamos o *solver* recomendado: *GLPK (GNU Linear Programming Kit)*.

4.1 Considerações iniciais

Pretendemos fazer o planeamento da produção de uma fábrica de plantas, que produz várias plantas para satisfazer a procura num horizonte dado. Dispomos de um número limitado de prateleiras que suporte, igualmente, um número limitado de paletes onde crescem as plantas. Além disso, as plantas têm tempos de crescimento distintos e requerem configurações específicas de água, luz, temperatura (por exemplo).

Na realidade, as fábricas não vendem frutas e legumes à unidade (avulso), mas sim em paletes (ou cabazes) com uma capacidade múltipla de um certo número inteiro. Suponhamos que fizemos uma encomenda de 34 unidades de alface. Em vez disso, se vender cabazes com 10 alfaces (por exemplo), a encomenda consistirá em 4 cabazes/paletes desse tipo.

A procura diária será dada em termos de um multiplicador m, isto é, assumindo que cada palete para venda tem capacidade para m unidades, a procura real num dia d será dada por $m \times p_d$.

Na fábrica de plantas assumimos que o movimento das paletes segue a ordem imposta pelas sucessivas fases desde a germinação até à etapa final do crescimento, de tal modo que o movimento recíproco entre fases de crescimento sucessivas não é permitido. Todas as prateleiras do mesmo tipo são agrupadas, e por isso, acomodam paletes com a mesma configuração, isto é, uma prateleira não pode ser ocupada por paletes que requerem configurações distintas. Para cada fase de crescimento i, denotamos por S_{T_i} o conjunto de prateleiras do tipo i que acomodam essa fase (ver figura 4.1).

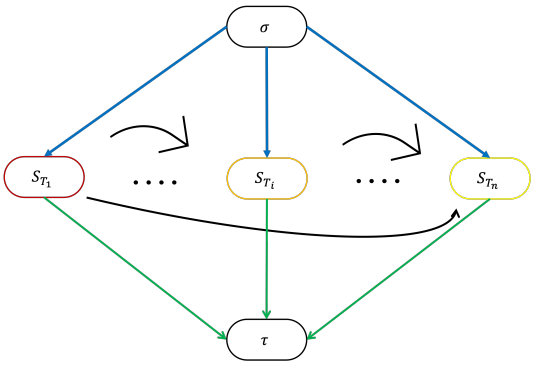


Figura 4.1: Trajeto das plantas na fábrica.

No contexto do problema MinM, as variáveis $y_{s,d,k}$ representam o estado da prateleira s no dia d na configuração k, e tomam valores no conjunto $\{0, 1\}$. Por outras palavras, se $y_{s,d,k} = 0$ então a prateleira s no dia d na configuração k está livre (ou também, não está em uso), caso contrário, está a ser utilizada.

A sementeira (σ) e o armazém (τ) serão tomadas como prateleiras, apesar das suas capacidades serem superiores às das prateleiras usadas para fazer crescer as plantas, mas não serão contabilizadas como tal. Assim, para cada caso precisamos de conhecer:

1. o período de laboração da fábrica ($\bar{d}$);
2. o número de plantas (*crops*) e a procura procura diária para cada uma delas;
3. o número de prateleiras, a distribuição pelos diferentes tipos, e a capacidade;
4. o número de configurações de crescimento;
5. o tempo e a sequência de configurações de crescimento para cada planta.

Para cada caso, apresentamos as restrições associadas, e as condições fronteira que envolvem sete condições de fixação de variáveis são referenciadas por FV1, FV2, ..., FV7.

4.2 Caso 1 – Procura num horizonte de 100 dias

Pretendemos fazer o planeamento da produção de uma fábrica de plantas, que produz alface para satisfazer a procura num horizonte de 100 dias, minimizando o número de movimentos. Dispomos de 10 prateleiras, cada uma com capacidade para 12 paletes. Supomos que a alface tem 3 fases de crescimento e demora 20 dias a crescer.

4.2.1 Dados do problema

Para a simulação computacional que vamos realizar, assumimos que:

1. horizonte de planeamento igual a $\bar{d} = 100$ dias ($D = \{1, \ldots, 99\}$);
2. tempo de crescimento de 20 dias ($\gamma = 20$, ou seja, $g \in \{0, \ldots, 20\}$);
3. conjunto das prateleiras, $S = \{s_1\,, s_2\,, s_3\,, s_4\,, s_5\,, s_6\,, s_7\,, s_8\,, s_9\,, s_{10}\}$;
4. conjunto das configurações, $K = \{I_1\,, I_2\,, I_3\}$;
5. conjunto dos tipos de prateleiras, $T = \{T_1\,, T_2\,, T_3\}$;
6. conjunto de prateleiras em cada tipo:
   - prateleiras do tipo T_1 : $S_{T_1} = \{s_1\,, s_2\,, s_3\}$;
   - prateleiras do tipo T_2 : $S_{T_2} = \{s_4\,, s_5\,, s_6\,, s_7\,, s_8\}$;
   - prateleiras do tipo T_3 : $S_{T_3} = \{s_9\,, s_{10}\}$.
7. todas as prateleiras admitem 12 paletes, independentemente das configurações, ou seja, $q_{s,\,k} = 12$, $\forall s \in S$, $\forall k \in K$;
8. encomenda como um vetor linha $[p_d]_{1 \leq d \leq 100}$;

Dia	1	...	23	24	25	26	27	...	48	49	50	...	100
Encomenda	0			5	8	12	0			18	0		

Tabela 4.1: Caracterização da encomenda em paletes.

9. duração e localização (tipos de prateleiras) adequada de cada fase de crescimento:

Nome	Germinação	Crescimento	Maturação
Duração (dias)	6	12	2
Prateleiras	S_{T_1}	S_{T_2}	S_{T_3}
Configuração	I_1	I_2	I_3

Tabela 4.2: Correspondência entre os conjuntos de prateleiras, as fases e configurações de crescimento.

Então, a sequência de configurações de crescimento, Kg, da alface é dada por: $\mathrm{Kg} = \{I_1\,, I_1\,, I_1\,, I_1\,, I_1\,, I_1\,, I_2\,, I_2\,, I_2\,, I_2\,, I_2\,, I_2\,, I_2\,, I_2\,, I_2\,, I_2\,, I_2\,, I_3\,, I_3\}$.

4.2.2 Modelação matemática

Neste caso, verificamos que $20 \times 99 \times 31 = 61380$ $[31 = (10^2 - 10)/2 - 14]$ variáveis de decisão entram na função objetivo. O número total de variáveis $x^g_{r,d,s}$ alocadas é $(20+1) \times 100 \times 64 = 134400$ $\left[64 = ((10+2)^2 - (10+2))/2 + 12 - 14\right]$ e o total das variáveis $y_{s,d,k}$ é de $10 \times (100-1) \times 3 = 2970$.

Na figura 4.2, está representado um grafo que descreve os movimentos possíveis das paletes entre as prateleiras de tipos/configurações diferentes. Cada aresta orientada representa um movimento de paletes possível. O gradiente de cor de cada aresta identifica a prateleira de saída de paletes e a prateleira de entrada delas. Por exemplo, o movimento de paletes entre prateleiras provenientes de S_{T_1} (cor castanha) e com destino S_{T_3} (cor azul) corresponde a aresta $S_{T_1} \longrightarrow S_{T_3}$.

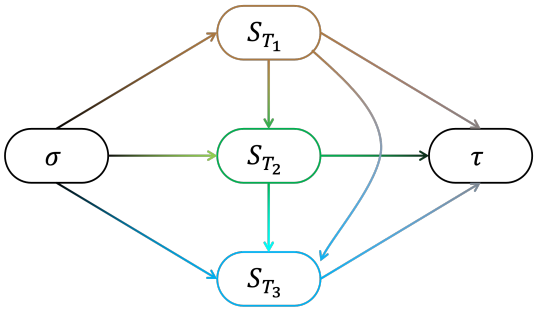


Figura 4.2: Trajeto esquemático dos movimentos possíveis das paletes de alface na fábrica.

Para facilitar a leitura, usamos as cores presentes na figura 4.2 para localizar e identificar cada prateleira. Apresentamos de seguida, todas as restrições associadas a este caso. O código desenvolvido é apresentado no Apêndice B.2.

4.2.2.1 Restrições do problema

Pretendemos minimizar o número de movimentos de paletes entre as prateleiras dos vários tipos, isto é, queremos:

$$\min \quad \sum_{g=1}^{20} \sum_{\substack{r,s \in S \\ r \neq s}} \sum_{d \in \{1,2,\ldots,99\}} x^{g}_{r,d,s} \,. \tag{4.1}$$

Restrições:

Procura:

$$\sum_{s \in T} x^{20}_{s,d-1,\tau} = p_d \quad \forall d \in D \cup \{\bar{d}\} = \{1, \ldots, 100\}$$

Capacidade:

$$\sum_{r \in S'} \sum_{\substack{g \in \{1,\ldots,20\} \\ k_g = k}} x^{g}_{s,d,r} \leq 12 \cdot y_{s,\,d,\,k} \quad \forall s \in S\,,\, \forall k \in K\,, \forall d \in \{1, 2, \ldots, 99\}$$

Plantação:

$$\sum_{s \in T} x^{0}_{\sigma,d-20-1,s} = p_d \quad \forall d \in \{22, \ldots, 100\}$$

Configuração:

$$\sum_{k \in K} \sum_{s \in S_t} y_{s,d,k} \leq n_t \quad \forall t \in T\,, \forall d \in D$$

Equilíbrio:

$$\sum_{r \in S'} x^{g}_{r,d-1,s} = \sum_{r \in S'} x^{g+1}_{s,d,r} \quad \forall g \in \{0, \ldots, 20\}\,, \forall s \in S\,, \forall d \in D$$

Fixação de variáveis:

FV1: $x^{g}_{t,d,\sigma} = x^{g}_{\tau,d,t} = x^{g}_{\sigma,d,\tau} = 0, \quad \forall t \in S', \forall g \in \{0, \ldots, 20\}, \forall d \in \{0, 1, \ldots, 99\}$

FV2:

$$\begin{aligned} x^{g}_{\sigma,d,t} &= 0 \qquad \forall t \in S', \forall g \in \{1, \ldots, 20\}, \forall d \in \{1, \ldots, 99\} \\ x^{g}_{t,d,\tau} &= 0 \qquad \forall t \in S', \forall g \in \{0, \ldots, 19\}, \forall d \in \{0, 1, \ldots, 99\} \\ x^{g}_{t_1,d,t_2} &= 0 \qquad \forall t_1,\, t_2 \in S, \forall g \in \{0,\, 20\}, \forall d \in \{1, \ldots, 99\} \end{aligned}$$

FV3: $\forall d \in \{0, 1, \ldots, 99\}$

$$x^0_{t_1,d,t_2} = 0 \qquad \forall\, t_1 \in S' \setminus \{\sigma\},\, \forall\, t_2 \in S'$$
$$x^{20}_{t_1,d,t_2} = 0 \qquad \forall\, t_1 \in S',\, \forall\, t_2 \in S' \setminus \{\tau\}$$

FV4: $d' = \max_{d\in\{1,\ldots,99\}}\{d :\, p_d > 0\}$ o último dia com procura. Temos $d' = 49$.

$x^g_{r,d,s} = 0 \qquad \forall s, r \in S,\, \forall d \in \{d' - 20 + 1, \ldots, d'\}, \forall g \in \{0, \ldots,\, 20 - (d' - d)\},$ ou seja,

$x^g_{r,d,s} = 0 \qquad \forall r, s \in S,\, \forall d \in \{30, 31, \ldots, 49\}, \forall g \in \{0, \ldots,\, d - 29\}$

FV5: $x^g_{r,d,s} = 0 \quad \forall r, s \in S',\, \forall d \in D' : d < 20,\, \forall g \in \{d + 1, \ldots, 20\}$

FV6:

- $x^0_{\sigma,d-20-1,r} \leq p_d \quad \forall r \in S,\, \forall d \in \{20 + 1, \ldots, 100\}$
- $x^{20}_{s,d-1,\tau} \leq p_d \quad \forall s \in S,\, \forall d \in \{1, 2, \ldots, 100\}$

FV7: $\eta_{d,\,k} \in \mathbb{N}$, o número de paletes que requerem a configuração k no dia d, temos

$$\eta_{d,\,k} = \sum_{d'=d+1}^{d+20} \sum_{k_{20-(d'-d-1)}=k} p_{d'} \quad \forall d \in \{1, \ldots, 99\},\, \forall k \in K.$$

Então, $\quad y_{s,d,k} = 0 \quad \forall s \in S,\, \forall d \in D,\, \forall k \in K : \eta_{d,\,k} = 0.$

4.2.2.2 Resultados da simulação

Vamos apresentar uma solução possível resultante da simulação computacional. A solução obtida não é única. Sendo assim:

1. para a encomenda do dia 24 temos:
   - $x^0_{\sigma,3,s_1} = 5$;
   - $x^1_{s_1,4,s_1} = x^2_{s_1,5,s_1} = x^3_{s_1,6,s_1} = x^4_{s_1,7,s_1} = x^5_{s_1,8,s_1} = x^6_{s_1,9,s_5} = 5$;
   - $x^7_{s_5,10,s_5} = x^8_{s_5,11,s_5} = x^9_{s_5,12,s_5} = x^{10}_{s_5,13,s_5} = x^{11}_{s_5,14,s_5} = x^{12}_{s_5,15,s_5} = 5$, $x^{13}_{s_5,16,s_5} = x^{14}_{s_5,17,s_5} = x^{15}_{s_5,18,s_5} = x^{16}_{s_5,19,s_5} = x^{17}_{s_5,20,s_5} = x^{18}_{s_5,21,s_{10}} = 5$;
   - $x^{19}_{s_{10},22,s_{10}} = x^{20}_{s_{10},23,\tau} = 5$.

2. para a encomenda do dia 25 temos:

   - $x^{0}_{\sigma,4,s_2} = 8$;
   - $x^{1}_{s_2,5,s_2} = x^{2}_{s_2,6,s_2} = x^{3}_{s_2,7,s_2} = x^{4}_{s_2,8,s_2} = x^{5}_{s_2,9,s_2} = x^{6}_{s_2,10,s_4} = 8$;
   - $x^{7}_{s_4,11,s_4} = x^{8}_{s_4,12,s_4} = x^{9}_{s_4,13,s_4} = x^{10}_{s_4,14,s_4} = x^{11}_{s_4,15,s_4} = x^{12}_{s_4,16,s_4} = 8$, $x^{13}_{s_4,17,s_4} = x^{14}_{s_4,18,s_4} = x^{15}_{s_4,19,s_4} = x^{16}_{s_4,20,s_4} = x^{17}_{s_4,21,s_4} = x^{18}_{s_4,22,s_9} = 8$;
   - $x^{19}_{s_9,23,s_9} = x^{20}_{s_9,24,\tau} = 8$.

3. para a encomenda do dia 26 temos:

   - $x^{0}_{\sigma,5,s_3} = 12$;
   - $x^{1}_{s_3,6,s_3} = x^{2}_{s_3,7,s_3} = x^{3}_{s_3,8,s_3} = x^{4}_{s_3,9,s_3} = x^{5}_{s_3,10,s_3} = x^{6}_{s_3,11,s_8} = 12$;
   - $x^{7}_{s_8,12,s_8} = x^{8}_{s_8,13,s_8} = x^{9}_{s_8,14,s_8} = x^{10}_{s_8,15,s_8} = x^{11}_{s_8,16,s_8} = x^{12}_{s_8,17,s_8} = 12$, $x^{13}_{s_8,18,s_8} = x^{14}_{s_8,19,s_8} = x^{15}_{s_8,20,s_8} = x^{16}_{s_8,21,s_8} = x^{17}_{s_8,22,s_8} = x^{18}_{s_8,23,s_{10}} = 12$;
   - $x^{19}_{s_{10},24,s_{10}} = x^{20}_{s_{10},25,\tau} = 12$.

4. para a encomenda do dia 49 temos:

   - $x^{0}_{\sigma,28,s_1} = 12$, $x^{0}_{\sigma,28,s_3} = 6$;
   - $x^{1}_{s_1,29,s_1} = x^{2}_{s_1,30,s_1} = x^{3}_{s_1,31,s_1} = x^{4}_{s_1,32,s_1} = x^{5}_{s_1,33,s_1} = x^{6}_{s_1,34,s_4} = 12$;
   - $x^{1}_{s_3,29,s_3} = x^{2}_{s_3,30,s_3} = x^{3}_{s_3,31,s_3} = x^{4}_{s_3,32,s_3} = x^{5}_{s_3,33,s_3} = x^{6}_{s_3,34,s_4} = 6$;
   - $x^{7}_{s_4,35,s_4} = x^{8}_{s_4,36,s_4} = x^{9}_{s_4,37,s_4} = x^{10}_{s_4,38,s_4} = x^{11}_{s_4,39,s_4} = x^{12}_{s_4,40,s_4} = 12$, $x^{13}_{s_4,41,s_4} = x^{14}_{s_4,42,s_4} = x^{15}_{s_4,43,s_4} = x^{16}_{s_4,44,s_4} = x^{17}_{s_4,45,s_4} = x^{18}_{s_4,46,s_9} = 12$;
   - $x^{7}_{s_5,35,s_5} = x^{8}_{s_5,36,s_5} = x^{9}_{s_5,37,s_5} = x^{10}_{s_5,38,s_5} = x^{11}_{s_5,39,s_5} = x^{12}_{s_5,40,s_5} = 6$, $x^{13}_{s_5,41,s_5} = x^{14}_{s_5,42,s_5} = x^{15}_{s_5,43,s_5} = x^{16}_{s_5,44,s_5} = x^{17}_{s_5,45,s_5} = x^{18}_{s_5,46,s_{10}} = 6$;
   - $x^{19}_{s_{10},47,s_9} = x^{20}_{s_9,48,\tau} = 12$;
   - $x^{19}_{s_{10},47,s_{10}} = x^{20}_{s_{10},48,\tau} = 6$.

Para melhor percebermos o significado real das variáveis indicadas acima vamos apresentá-las de forma esquemática nas Tabelas 4.3 e 4.4, e para facilitar a análise e leitura, cada um dos símbolos (5) , (6) , (8) , (12) , (18) , representa, respetivamente, $5, 6, 8, 12$ e 18 paletes de alface.

Nas duas tabelas, as duas linhas iniciais têm a identificação das prateleiras, onde prateleiras do mesmo tipo partilham a mesma cor, e para simplificar, apresentamos apenas os dias e prateleiras relevantes para o planeamento.

		S_{T_1}			S_{T_2}			S_{T_3}		
Dia	σ	s_1	s_2	s_3	s_4	s_5	s_8	s_9	s_{10}	τ
3	5									
4	8	5								
5	12	5	8							
6		5	8	12						
7		5	8	12						
8		5	8	12						
9		5	8	12						
10			8	12		5				
11				12	8	5				
12					8	5	12			
13					8	5	12			
14					8	5	12			
15					8	5	12			
16					8	5	12			
17					8	5	12			
18					8	5	12			
19					8	5	12			
20					8	5	12			
21					8	5	12			
22					8		12		5	
23							12	8	5	
24								8	12	5
25									12	8
26										12
27										

Tabela 4.3: Planeamento da produção de alface para as entregas das encomendas para o 24ž, 25ž e 26ž dias de laboração da fábrica.

		S_{T_1}		S_{T_2}		S_{T_3}		
Dia	σ	s_1	s_3	s_4	s_5	s_9	s_{10}	τ
28	18							
29		12	6					
30		12	6					
31		12	6					
32		12	6					
33		12	6					
34		12	6					
35				12	6			
36				12	6			
37				12	6			
38				12	6			
39				12	6			
40				12	6			
41				12	6			
42				12	6			
43				12	6			
44				12	6			
45				12	6			
46				12	6			
47						12	6	
48						12	6	
49								18

Tabela 4.4: Planeamento da produção de alface para as entregas das encomendas para o $49ž$ dia de laboração da fábrica.

Os valores das encomendas dos dias 24 e dia 25 são inferiores à capacidade máxima de cada prateleira, mas a sua soma supera essa capacidade. Por isso, nenhuma prateleira pode acomodar simultaneamente paletes das duas encomendas. Então, a sequência de movimentos de paletes:

- para a encomenda do dia 24 (cor roxa) é $\sigma \to s_1 \to s_5 \longrightarrow s_{10} \to \tau$;
- para a encomenda do dia 25 (cor cinza) é $\sigma \to s_2 \longrightarrow s_4 \longrightarrow s_9 \to \tau$.

Como cada prateleira acomoda, no máximo, 12 paletes de alface, da Tabela 4.3, podemos verificar que todas as prateleiras atingem a sua capacidade quando recebem paletes para satisfazer a encomenda do dia 26 (cor laranja) estavam cheias. Por último, a sequência de movimentos obtida para satisfazer a encomenda é $\sigma \to s_3 \longrightarrow s_8 \longrightarrow s_{10} \to \tau$.

Para a encomenda do dia 49, notamos que o seu valor é superior à capacidade máxima de cada prateleira, e por isso, é necessário distribuir as paletes por mais do que uma prateleira do mesmo tipo. Desse modo, cada conjunto de paletes percorrerá uma sequência de movimentos distinta, mas cujo total seja o valor pedido de 18 paletes de alface.

Das 18 paletes, 12 seguem a sequência (apresentada na Tabela 4.4 e na figura 4.3 com a cor laranja): $\sigma \to s_1 \longrightarrow s_4 \longrightarrow s_9 \to \tau$. As restantes 6 seguem (indicada na Tabela 4.4 e na figura 4.3 com a cor verde): $\sigma \to s_3 \longrightarrow s_5 \longrightarrow s_{10} \to \tau$.

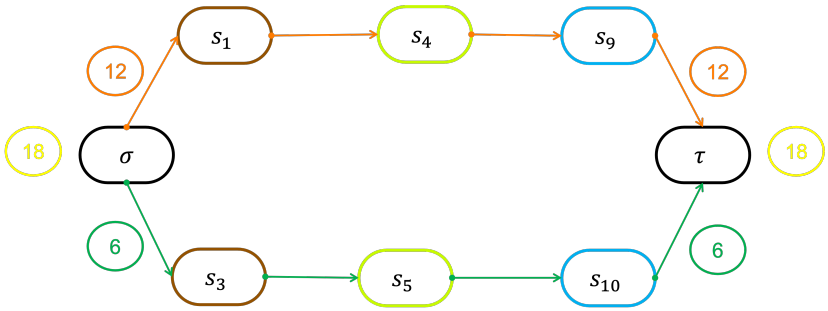


Figura 4.3: Trajetos na fábrica das 18 paletes, divididas em dois conjuntos com 6 e 12 paletes de alface na fábrica, para satisfazer a encomenda no dia 49.

Indicamos agora todas as variáveis y que não são zero:

- $y_{s_1,d,I_1} = 1,\ \forall d \in \{4,\dots,9,29,\dots,34\}$;
- $y_{s_2,d,I_1} = 1,\ \forall d \in \{5,\dots,10\}$;
- $y_{s_3,d,I_1} = 1,\ \forall d \in \{6,\dots,11,29,\dots,34\}$;
- $y_{s_4,d,I_2} = 1,\ \forall d \in \{11,\dots,22,35,\dots,46\}$;
- $y_{s_5,d,I_2} = 1,\ \forall d \in \{10,\dots,23,35,\dots,46\}$;
- $y_{s_8,d,I_2} = 1,\ \forall d \in \{12,\dots,23\}$;

- $y_{s_9,d,I_3} = 1,\ \forall d \in \{23, 24, 47, 48\}$;
- $y_{s_{10},d,I_3} = 1,\ \forall d \in \{22, 23, 24, 25, 47, 48\}$.

4.3 Caso 2 – Procura de duas plantas diferentes

Para perceber as diferenças entre o modelo de planeamento com uma só planta e outro com mais do que uma, vamos considerar um caso simples.

Tal como no caso anterior, pretendemos planear a produção de uma fábrica de plantas, que produz cebolinho e ervilhas, para satisfazer a procura num horizonte de 15 dias, minimizando o número de movimentos. Uma vez que, na ervilheira, cada vagem contem mais do que uma ervilha, as encomendas são feitas em paletes de pés de ervilhas/cebolinho. Dispomos de 16 prateleiras, cada uma com capacidade para 5 paletes. Supomos ainda que, o cebolinho tem 4 fases de crescimento e que demora 8 dias a crescer, e as ervilhas têm 5 fases de crescimento e demora 10 dias a crescer[1].

4.3.1 Dados do problema

Para a simulação computacional que vamos realizar, assumimos que:

1. conjunto das plantas: $C = [\text{cebolinho, ervilhas}] = [c_1,\ c_2]$;
2. horizonte de planeamento igual a $\bar{d} = 15$ dias ($D = \{1, \ldots, 14\}$);
3. tempo de crescimento do cebolinho $\gamma_{c_1} = 8$ dias ($\gamma_{c_1} = 8$, logo, $g_{c_1} \in \{0, \ldots, 8\}$), e o tempo de crescimento das ervilhas $\gamma_{c_2} = 10$ dias ($\gamma_{c_2} = 10$, logo, $g_{c_2} \in \{0, \ldots, 10\}$) ;
4. conjunto das prateleiras, $S = \{s_1\,, s_2\,, s_3\,, \ldots\,, s_{14}\,, s_{15}\,, s_{16}\}$;
5. conjunto das configurações, $K = \{I_1\,, I_2\,, I_3\,, I_4\,, I_5\}$;
6. conjunto dos tipos de prateleiras, $T = \{T_1\,, T_2\,, T_3\,, T_4\,, T_5\}$;
7. conjunto das prateleiras em cada tipo:
   - prateleiras do tipo $T_1:\ S_{T_1} = \{s_1\,, s_2\,, s_3\}$;
   - prateleiras do tipo $T_2:\ S_{T_2} = \{s_4\,, s_5\,, s_6\,, s_7\}$;

[1] Na realidade, é pouco provável que o cebolinho e as ervilhas cresçam num período tão curto.

- prateleiras do tipo T_3 : $S_{T_3} = \{s_8\,, s_9\,, s_{10}\,, s_{11}\}$;
- prateleiras do tipo T_4 : $S_{T_4} = \{s_{12}\,, s_{13}\,, s_{14}\}$;
- prateleiras do tipo T_5 : $S_{T_5} = \{s_{15}\,, s_{16}\}$.

8. a capacidade de cada prateleira é limitada e é igual para todas qualquer que seja a configuração, isto é, $q_{s,\,k} = 5$, $\forall s \in S$, $\forall k \in K$;

9. encomenda como uma matriz $(c \times d)$: $[p_{c,d}]_{\left(c \in C,\ d \in D \cup \{\bar{d}\}\right)}$;

		Dia	1	...	11	12	13	14	15
Encomenda	c_1	cebolinho	0			2	0		
	c_2	ervilhas	0				3	0	

Tabela 4.5: Caracterização da encomenda para as duas plantas em função do número de paletes pretendidas.

10. duração e localização (tipo de prateleiras) adequada de cada fase de crescimento (a ordem das fases de crescimento do cebolinho é C1, C2, C3, C4, e a ordem para as ervilhas é E1, E2, E3, E4, E5);

	Cebolinho				Ervilhas				
Fase de crescimento	C1	C2	C3	C4	E1	E2	E3	E4	E5
Duração (dias)	1	2	4	1	2	3	3	1	1
Configuração	I_1	I_2	I_3	I_4	I_1	I_2	I_3	I_4	I_5
Prateleiras	S_{T_1}	S_{T_2}	S_{T_3}	S_{T_4}	S_{T_1}	S_{T_2}	S_{T_3}	S_{T_4}	S_{T_5}

Tabela 4.6: Correspondência entre as fases, durações e configurações de crescimento de cada planta, e a localização nas prateleiras. Fases de crescimento do cebolinho: C1, C2, C3, C4; Fases de crescimento das ervilhas: E1, E2, E3, E4, E5.

Assim, a sucessão de configurações para cada planta é dada por:

- cebolinho: $\mathrm{Kg}_{c_1} = \{I_1\,, I_2\,, I_2\,, I_3\,, I_3\,, I_3\,, I_3\,, I_4\}$;
- ervilhas: $\mathrm{Kg}_{c_2} = \{I_1\,, I_1\,, I_2\,, I_2\,, I_2\,, I_3\,, I_3\,, I_3\,, I_4\,, I_5\}$.

4.3.2 Modelação matemática

Para este caso, verificamos que entram na função objetivo $(8 + 10) \times 14 \times 101 = 25452$ ($101 = (16^2 - 16)/2 - 19$) variáveis de decisão. O número total de variáveis $x_{r,d,s}^{c,g}$ alocadas no

computador é $[(8+1)+(10+1)]\times 152\times 15 = 45600$ $\left(152 = ((16+2)^2 - (16+2))/2 + 18 - 19\right)$, e as variáveis $y_{s,d,k}$ totalizam $16 \times (15-1) \times 5 = 1120$.

Na figura 4.4 está representado um grafo que descreve os movimentos possíveis das paletes entre as prateleiras dos diferentes tipos/ configurações. Cada aresta orientada representa um movimento de paletes possível, e a sua cor identifica a prateleira de saída de paletes e a prateleira de entrada delas. Exemplificando, o movimento de paletes entre prateleiras provenientes de S_{T_2} (cor amarela) e com destino S_{T_5} (cor roxa) corresponde a aresta $S_{T_2} \longrightarrow S_{T_5}$.

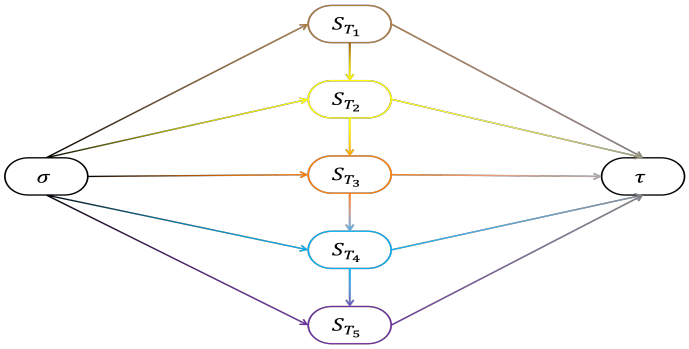


Figura 4.4: Trajeto esquemático dos movimentos possíveis das paletes na fábrica.

Para facilitar a leitura, as cores que usamos na figura 4.4 servem para localizar e identificar cada prateleira no conjunto a que pertence. De seguida, exibimos todas as restrições associadas a este caso. O código desenvolvido é apresentado no Apêndice B.3.

4.3.2.1 Função objetivo e restrições do problema

Pretendemos minimizar o número de movimentos de paletes entre as prateleiras dos vários tipos, ou seja, queremos:

$$\min \sum_{c\in C} \sum_{g=1}^{\gamma_c} \sum_{\substack{r,s\in S \\ r\neq s}} \sum_{d\in D} x_{r,d,s}^{c,g}\,. \tag{4.2}$$

Expandindo a função objetivo 4.2, obtemos:

$$\min \sum_{d\in\{1,\ldots,14\}} \sum_{\substack{r,\,s\in S \\ r\neq s}} \left(\sum_{g=1}^{8} x_{r,d,s}^{c_1,g} + \sum_{g=1}^{10} x_{r,d,s}^{c_2,g} \right). \tag{4.3}$$

Restrições:

Procura: $\forall d \in D \cup \{\bar{d}\} = \{1, \ldots, 15\}$:

$$\sum_{s \in S} x^{c_1,8}_{s,d-1,\tau} = p_{c_1,d} \qquad \sum_{s \in S} x^{c_2,10}_{s,d-1,\tau} = p_{c_2,d}$$

Capacidade: $\forall d \in \{1, \ldots, 14\}$, $\forall k \in K$:

$$\sum_{r \in S'} \left(\sum_{\substack{g \in \{1,\ldots,8\} \\ k = k_{c_1,\,g}}} x^{c_1,g}_{s,d,r} + \sum_{\substack{g \in \{1,\ldots,10\} \\ k = k_{c_2,\,g}}} x^{c_2,g}_{s,d,r} \right) \leq q_{s,\,k} \cdot y_{s,d,k}\,, \qquad \forall\, s \in S$$

Plantação:

$$\sum_{s \in S} x^{c_1,0}_{\sigma,d-8-1,s} = p_{c_1,d} \quad \forall\, d \in \{8+1, \ldots, 15\}$$
$$\sum_{s \in S} x^{c_2,0}_{\sigma,d-10-1,s} = p_{c_2,d} \quad \forall\, d \in \{10+1, \ldots, 15\}$$

Equilíbrio:

$$\sum_{r \in S'} x^{c_1,g}_{r,d-1,s} = \sum_{r \in S'} x^{c_1,g+1}_{s,d,r} \quad \forall\, g \in \{0, \ldots, 8\},\, \forall\, s \in S,\, \forall\, d \in \{1, \ldots, 14\}$$
$$\sum_{r \in S'} x^{c_2,g}_{r,d-1,s} = \sum_{r \in S'} x^{c_2,g+1}_{s,d,r} \quad \forall\, g \in \{0, \ldots, 10\},\, \forall\, s \in S,\, \forall\, d \in \{1, \ldots, 14\}$$

Configuração: $\forall\, d \in \{1, 2, \ldots, 14\}$:

$$\sum_{k \in K} \sum_{s \in S_t} y_{s,d,k} \leq n_t \quad \forall\, t \in T$$

Fixação de variáveis:

FV1: $\forall\, d \in \{0, 1, \ldots, 14\}$:

$$x^{c_1,g}_{s,d,\sigma} = x^{c_1,g}_{\tau,d,s} = x^{c_1,g}_{\sigma,d,\tau} = 0, \qquad \forall\, s \in S',\, \forall g \in \{0, \ldots, 8\}$$
$$x^{c_2,g}_{s,d,\sigma} = x^{c_2,g}_{\tau,d,s} = x^{c_2,g}_{\sigma,d,\tau} = 0, \qquad \forall\, s \in S',\, \forall g \in \{0, \ldots, 10\}$$

FV2:

$$\begin{aligned}
\forall\, d \in \{0,1,\ldots,14\},\, \forall\, s \in S' : \quad & x^{c_1,g}_{\sigma,d,s} = 0, \quad \forall\, g \in \{1,\ldots,8\} \\
& x^{c_2,g}_{\sigma,d,s} = 0, \quad \forall\, g \in \{1,\ldots,10\} \\
\forall\, d \in \{0,1,\ldots,14\},\, \forall\, s \in S' : \quad & x^{c_1,g}_{s,d,\tau} = 0, \quad \forall\, g \in \{0,1,\ldots,8-1\} \\
& x^{c_2,g}_{s,d,\tau} = 0, \quad \forall\, g \in \{0,1,\ldots,10-1\} \\
\forall\, d \in \{1,\ldots,14\},\, \forall\, r,s \in S : \quad & x^{c_1,g}_{r,d,s} = 0, \quad \forall\, g \in \{0,8\} \\
& x^{c_2,g}_{r,d,s} = 0, \quad \forall\, g \in \{0,10\}
\end{aligned}$$

FV3: $\forall d \in \{0,1,\ldots,14\}$:

$$\begin{aligned}
\forall\, r \in S' \setminus \{\sigma\},\, \forall s \in S' : \quad & x^{c_1,0}_{r,d,s} = x^{c_2,0}_{r,d,s} = 0 \\
\forall\, r \in S',\, \forall s \in S' \setminus \{\tau\} : \quad & x^{c_1,8}_{r,d,s} = x^{c_2,10}_{r,d,s} = 0
\end{aligned}$$

FV4: $d'_c = \max_{d\in\{1,\ldots,14\}}\{d : p_{c,\,d} > 0\}$ o último dia com procura da planta $c \in C$. Temos $d'_{c_1} = 12\,, d'_{c_2} = 13$, e:

$$\begin{aligned}
& x^{c_1,g}_{r,d,s} = 0, \quad \forall\, r,s \in S',\, \forall\, d \in \{12-8+1,\ldots,12\},\, \forall\, g \in \{0,\ldots,8-(12-d)\} \\
& x^{c_2,g}_{r,d,s} = 0, \quad \forall\, r,s \in S',\, \forall\, d \in \{13-10+1,\ldots,13\},\, \forall\, g \in \{0,\ldots,10-(13-d)\}
\end{aligned}$$

que são equivalentes, respetivamente, a:

$$\begin{aligned}
& x^{c_1,g}_{r,d,s} = 0, \quad \forall\, r,s \in S',\, \forall\, d \in \{5,\ldots,12\},\, \forall\, g \in \{0,\ldots,d-4\}. \\
& x^{c_2,g}_{r,d,s} = 0, \quad \forall\, r,s \in S',\, \forall\, d \in \{4,\ldots,13\},\, \forall\, g \in \{0,\ldots,d-3\}.
\end{aligned}$$

FV5: $\forall\, r,s \in S'$:

$$\begin{aligned}
& x^{c_1,g}_{r,d,s} = 0, \quad \forall\, d \in \{0,1,\ldots,14\} :\, d < 8,\, \forall\, g \in \{d+1,\ldots,8\} \\
& x^{c_2,g}_{r,d,s} = 0, \quad \forall\, d \in \{0,1,\ldots,14\} :\, d < 10,\, \forall\, g \in \{d+1,\ldots,10\}
\end{aligned}$$

FV6:

$$\forall\, s \in S : \quad x^{c_1,0}_{\sigma,d-8,s} \leq p_{c_1\,,d}\ \ \forall\, d \in \{8+1,\ldots,15\}$$

$$
\begin{aligned}
& x^{c_2,0}_{\sigma,d-10,s} \leq p_{c_2,d} \;\; \forall\, d \in \{10+1,\ldots,15\} \\
\forall\, s \in S: \quad & x^{c_1,8}_{s,d-1,\tau} \leq p_{c_1,d} \;\; \forall\, d \in \{1,\ldots,15\} \\
& x^{c_2,10}_{s,d-1,\tau} \leq p_{c_2,d} \;\; \forall\, d \in \{1,\ldots,15\}
\end{aligned}
$$

FV7: $\eta_{d,k} \in \mathbb{N}$, o número de paletes que requerem a configuração k no dia d, temos

$$
\eta_{d,k} = \sum_{d'=d+1}^{d+\gamma_c} \sum_{\substack{c \in C: \\ k_{c,\,\gamma_c-(d'-d-1)}=k}} p_{d'} \quad \forall d \in \{1,\ldots,14\},\, \forall k \in K,
$$

que podemos reescrever obtendo, $\forall d \in \{1,\ldots,14\}$, $\forall k \in K$:

$$
\eta_{d,k} = \sum_{d'=d+1}^{d+8} \sum_{k_{c_1\,,\,8-(d'-d-1)}=k} p_{c_1,d'} \; + \sum_{d'=d+1}^{d+10} \sum_{k_{c_2\,,\,10-(d'-d-1)}=k} p_{c_2,d'} \; .
$$

Assim, $y_{s,d,k} = 0$, $\forall s \in S$, $\forall d \in \{1,2,\ldots,14\}$, $\forall k \in K : \eta_{d,k} = 0$.

4.3.2.2 Resultados da simulação

De seguida, apresentamos uma das soluções possíveis resultante da implementação computacional. Indicamos as variáveis com valor não nulo.

1. para a encomenda do dia 12, constituída por 2 paletes de pés de cebolinho (c_1), obtivemos:

   - $x^{c_1,0}_{\sigma,3,s_3} = x^{c_1,1}_{s_3,4,s_7} = x^{c_1,2}_{s_7,5,s_7} = 2$;
   - $x^{c_1,3}_{s_7,6,s_{11}} = x^{c_1,4}_{s_{11},7,s_{11}} = x^{c_1,5}_{s_{11},8,s_{11}} = x^{c_1,6}_{s_{11},9,s_{11}} = 2$;
   - $x^{c_1,7}_{s_{11},10,s_{14}} = x^{c_1,8}_{s_{14},11,\tau} = 2$.

2. para a encomenda do dia 13, que consiste em 3 paletes de pés de ervilhas (c_2), obtivemos:

   - $x^{c_2,0}_{\sigma,2,s_3} = x^{c_2,1}_{s_3,3,s_3} = x^{c_2,2}_{s_3,4,s_7} = 3$;
   - $x^{c_2,3}_{s_7,5,s_7} = x^{c_2,4}_{s_7,6,s_7} = x^{c_2,5}_{s_7,7,s_9} = 3$;
   - $x^{c_2,6}_{s_9,8,s_9} = x^{c_2,7}_{s_9,9,s_9} = x^{c_2,8}_{s_9,10,s_{14}} = 3$;
   - $x^{c_2,9}_{s_{14},11,s_{16}} = x^{c_2,10}_{s_{16},12,\tau} = 3$.

Para entendermos melhor o valor que cada variável toma, elaboramos uma tabela onde estão expressos os movimentos das paletes de cebolinho e ervilhas para cada dia (ver Tabela 4.7). Para distinguir as plantas usaremos o símbolo (2) para representar as duas paletes de pés de cebolinho, e (3) para representar as três paletes de pés de ervilhas.

		S_{T_1}	S_{T_2}	S_{T_3}		S_{T_4}	S_{T_5}	
Dias	σ	s_3	s_7	s_9	s_{11}	s_{14}	s_{16}	τ
1								
2	(3)							
3	(2)	(3)						
4		(2)						
		(3)						
5			(2)					
			(3)					
6			(2)					
			(3)					
7			(3)		(2)			
8				(3)	(2)			
9				(3)	(2)			
10				(3)	(2)			
11						(2)		
						(3)		
12							(3)	(2)
13								(3)
14								
15								

Tabela 4.7: Planeamento da produção de cebolinho e ervilhas para as entregas das encomendas para o $12ž$ e $13ž$, respetivamente, dias de laboração da fábrica.

Por análise das variáveis, podemos notar que nos dias 4, 5, 6 e 11, as prateleiras que estão em uso acomodam paletes de ambas as plantas, e é notória a tendência de usar o mínimo de prateleiras. Por exemplo, uma outra solução sugere que as paletes de ervilhas nunca ocupem, simultaneamente, o mesmo tipo de prateleiras que as paletes de cebolinho.

Assim, a sequência de movimentos das paletes de cebolinho (percurso verde da figura 4.5) é dada por $\sigma \to s_3 \longrightarrow s_7 \longrightarrow s_9 \longrightarrow s_{14} \to \tau$; já a sequência para as paletes de ervilhas (percurso cinza da figura 4.5) é $\sigma \to s_3 \longrightarrow s_7 \longrightarrow s_{11} \longrightarrow s_{14} \longrightarrow s_{16} \to \tau$.

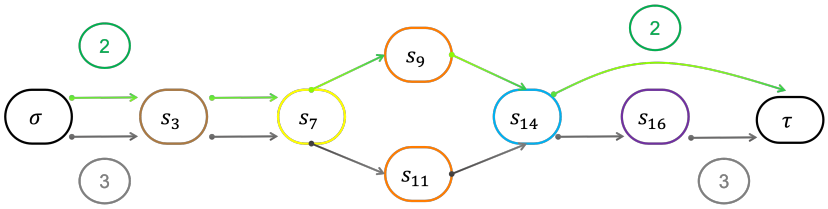


Figura 4.5: Trajetos na fábrica das 2 paletes de cebolinho, e das 3 paletes de ervilhas.

Indicamos agora todas as variáveis y que não são zero:

- $y_{s_3,3,I_1} = y_{s_3,4,I_1} = 1$;
- $y_{s_7,5,I_2} = y_{s_7,6,I_2} = y_{s_7,7,I_2} = 1$;
- $y_{s_9,8,I_3} = y_{s_9,9,I_3} = y_{s_9,10,I_3} = 1$;
- $y_{s_{11},7,I_3} = y_{s_{11},8,I_3} = y_{s_{11},9,I_3} = y_{s_{11},10,I_3} = 1$;
- $y_{s_{14},11,I_4} = y_{s_{16},12,I_4} = 1$.

4.4 Análise dos resultados

Neste capítulo, pretendíamos minimizar o número de movimentos em dois casos: planeamento de encomendas de duas plantas diferentes num curto período, e planeamento de encomendas de um só tipo de plantas, mas num período longo.

Em ambos os casos, o facto de haver mais do que uma prateleira de cada tipo conduz a existência de múltiplas soluções. Ainda assim, se no caso 1 (Secção 4.2) mantivéssemos todos os dados, mas diminuíssemos a capacidade máxima das prateleiras, estaríamos a limitar o número de soluções, e também a viabilidade do problema.

Podemos ainda notar que fixando a capacidade máxima (que assumimos ser igual para todas as prateleiras, independentemente da configuração em que se encontram), o valor máximo que pode ser encomendado num só dia está limitado pela capacidade total do conjunto $S_t\,, t \in T,$ com o menor número de prateleiras, – no caso 1, o conjunto S_{T_3} é constituído pelo menor número de prateleiras (s_9 e s_{10}) cuja capacidade total é de 24 paletes; logo a maior encomenda possível para um dia é de 24 paletes de alface.

Para simplificação do modelo, assumimos que não se podiam mover paletes entre prateleiras do mesmo tipo. Com isso, se planearmos a produção de uma planta, impedimos a movimentação desnecessária de paletes entre prateleiras do mesmo tipo. Por outro lado, se tivermos uma carteira de encomendas de várias plantas ou mesmo só de uma, a movimentação de paletes entre prateleiras do mesmo tipo revela-se vantajosa.

Permitindo essa movimentação, ao mudar as paletes necessárias para prateleiras do mesmo tipo/configuração que não estejam completamente cheias, por forma a que estas estejam na capacidade máxima, conseguimos rentabilizar e melhorar a produção.

Por último, vamos comparar vários parâmetros dos dois casos abordados neste capítulo com os do caso simples (Secção 3.3), que são indicados na Tabela 4.8.

		Casos		
	Parâmetros	Caso simples	Caso 1	Caso 2
	Plantas (C)	1	1	2
	Dias ($\bar{d}$)	10	100	15
	Configurações (K)	4	3	5
	Tempo de crescimento (γ_c)	6	20	8
				10
	Prateleiras (S)	4	10	16
	Tipos de prateleiras (T)	4	3	5
Total de variáveis: $x^{c,g}_{r,d,s}$ e $y_{s,d,k}$	Sem imposições [2]	2664	305370	98320
	Com imposições	1614	137370	46720
	Total de restrições	3009	118736	55858
	Tempo de execução (s)	≈ 0.05	≈ 2.73	≈ 0.74

Tabela 4.8: Comparação dos parâmetros dos casos simulados.

Com as hipóteses que impusemos na movimentação das paletes e na configuração das prateleiras, o número de variáveis $x^{c,g}_{r,d,s}$ é mais baixo do que sem essas hipóteses.

Como seria de esperar, um período de planeamento maior requer um elevado número de variáveis e, com isso, a execução é mais demorada. No caso 2, apesar de termos procura de duas plantas num curto período de planeamento, o tempo de execução é mais baixo do que no caso 1, mas o número de variáveis é bastante elevado quando comparado com

[2] Sem as imposições feitas na Secção 4.1.

o caso análogo para uma só planta.

Para concluir, os tempos de execução dos três casos são baixos. Apesar disso, o código/implementação desenvolvido pode ser melhorado no sentido de incorporar um modo eficiente de inserção das enumeras restrições associadas a este tipo de problema.

Capítulo 5

Conclusões

O propósito desta dissertação foi o estudo de modelos de planeamento de encomendas numa fábrica vertical de plantas. Uma exposição inicial foi essencial para localizar bibliográfica e cronologicamente os modelos já existentes: contexto de aplicabilidade, metodologia e eficácia.

Focámo-nos no artigo «The crop growth planning problem in vertical farming» (Santini et al. 2021), que permitiu explorar modelos de planeamento com características distintas dos modelos de planeamento já existentes – a particularidade de o crescimento das plantas não poder ser suspenso e retomado noutro momento.

Particularizamos duas situações numa fábrica com um objetivo – minimizar o número de movimentos –, e fizemos a sua implementação em *Python*. As ferramentas do pacote *Pyomo* permitiram uma boa manipulação das variáveis e das restrições, principalmente, na sua introdução.

Os resultados obtidos foram coerentes com o que se esperava, ainda que a implementação computacional não seja bastante eficiente.

Os modelos abordados são recentes, complexos, e requerem tempo para a sua completa compreensão e possível aplicação noutras áreas.

5.1 Trabalho futuro

São muitos os modelos de planeamento de tarefas, mas que não contemplam características como os modelos de Santini et al. 2021. Os pontos que os distinguem poderão ser a chave para as suas aplicações noutras áreas.

Para as simulações que realizamos, impossibilitamos os movimentos de paletes entre prateleiras que estão no mesmo tipo/configuração. Seria importante analisar a situação resultante caso fosse possível essa movimentação num contexto geral.

Nesta dissertação, assumimos, implicitamente, que os movimentos das paletes entre as prateleiras eram realizados pelos trabalhadores. Sendo assim, seria relevante estudar a relação entre o planeamento de encomendas numa fábrica e o horário dos trabalhadores, de forma a que haja trabalhadores suficientes nas etapas que requerem a movimentação das paletes e que seja otimizada a produção.

Bibliografia

Adenso-Díaz, Belarmino e Gabriel Villa (29 de set. de 2021). «Crop Planning in Synchronized Crop-Demand Scenarios: A Biobjective Optimization Formulation». Em: *Horticulturae* 7.10, p. 347. ISSN: 2311-7524. DOI: 10.3390/horticulturae7100347. URL: https://www.mdpi.com/2311-7524/7/10/347.

Al-Kodmany, Kheir (2018). «The Vertical Farm: A Review of Developments and Implications for the Vertical City». Em: *Buildings* 8.2. ISSN: 2075-5309. DOI: 10.3390/buildings8020024. URL: https://www.mdpi.com/2075-5309/8/2/24.

Benke, Kurt e Bruce Tomkins (jan. de 2017). «Future food-production systems: vertical farming and controlled-environment agriculture». Em: *Sustainability: Science, Practice and Policy* 13.1, pp. 13–26. ISSN: 1548-7733. DOI: 10.1080/15487733.2017.1394054. URL: https://www.tandfonline.com/doi/full/10.1080/15487733.2017.1394054.

Bennell, Julia, Toni Martinez e Chris Potts (2017). «Scheduling for the growing of crops to meet demand». Em: *MIC17*, p. 1.

Bessonov, N., V. Volpert e V.A. Volpert (2006). *Dynamic models of plant growth*. EPU, Éditions Publibook université. Publibook. ISBN: 9782748331653.

Brandon, Merril F. et al. (2016). «The Next Evolution of Agriculture: A Review of Innovations in Plant Factories». Em: *Handbook of Photosynthesis*. Ed. por Mohammad Pessarakli. 3ª ed. CRC Press.

Carrajola, Cristina, Maria José Castro e Teresa Hilário (2007). *Planeta com Vida 10 (Biologia)*. 1ª ed. Santillana, p. 89.

Despommier, Dickson (2013). «Farming up the city: the rise of urban vertical farms». Em: *Trends in Biotechnology* 31.7, pp. 388–389. ISSN: 0167-7799. DOI: https://doi.org/10.1016/j.tibtech.2013.03.008. URL: https://www.sciencedirect.com/science/article/pii/S016777991300070X.

Glen, John J. (1987). «Mathematical Models in Farm Planning: A Survey». Em: *Operations Research* 35.5, pp. 641–666. URL: http://www.jstor.org/stable/171218.

Hart, W.E. et al. (2021). *Pyomo Optimization Modeling in Python*. 3ª ed. Springer Optimization and Its Applications. Springer International Publishing. ISBN: 9783319588216.

Johnson, Kenneth A. Mason; Jonathan B. Losos; Susan R. Ringer; Peter H. Raven; George B. (2010). *Biology, 9th Edition*. 9ª ed. McGraw-Hill. ISBN: 0078936497; 9780078936494.

Kozai, T. (jan. de 2013). «Plant Factory in Japan - Current situation and perspectives». Em: *Chronica horticulturae* 53, pp. 8–11.

Kozai, Toyoki, Genhua Niu e Michiko Takagaki, eds. (2019). *Plant factory: an indoor vertical farming system for efficient quality food production*. 2ª ed. Cambridge: Elsevier. ISBN: 978-0-12-816691-8.

Li, Kai e Shan-lin Yang (abr. de 2009). «Non-identical parallel-machine scheduling research with minimizing total weighted completion times: Models, relaxations and algorithms». Em: *Applied Mathematical Modelling* 33.4, pp. 2145–2158. ISSN: 0307904X. DOI: 10.1016/j.apm.2008.05.019. URL: https://linkinghub.elsevier.com/retrieve/pii/S0307904X08001364.

Naskali, A. Teoman, Ozgun Pinarer e A. Cagri Tolga (2021). «Vertical Farming: Under Climate Change Effect». Em: *Environment and Climate-smart Food Production*. Ed. por Charis M. Galanakis. Cham: Springer International Publishing, pp. 259–284. ISBN: 978-3-030-71571-7. DOI: 10.1007/978-3-030-71571-7_8. URL: https://doi.org/10.1007/978-3-030-71571-7_8.

Nelson, D.L., R.D. Nelson e M.M. Cox (2004). *Lehninger Principles of Biochemistry, Fourth Edition*. W.H. Freeman. ISBN: 9780716764380.

Postlethwait, J.H. e J.L. Hopson (2006). *Modern Biology*. Modern Biology. Houghton Mifflin School. ISBN: 9780030651786.

Roose T.; Schnepf, A. (set. de 2008). «Mathematical models of plant-soil interaction». Em: *Philosophical Transactions Mathematical Physical and Engineering Sciences* 366 (1885), pp. 4597–4611. DOI: 10.1098/rsta.2008.0198.

Santini, Alberto et al. (out. de 2021). «The crop growth planning problem in vertical farming». Em: *European Journal of Operational Research* 294.1, pp. 377–390. ISSN: 03772217. DOI: 10.1016/j.ejor.2021.01.034. URL: https://linkinghub.elsevier.com/retrieve/pii/S0377221721000643.

Shamshiri, Redmond et al. (jan. de 2018). «Advances in greenhouse automation and controlled environment agriculture: A transition to plant factories and urban agriculture». Em: *International Journal of Agricultural and Biological Engineering* 11. DOI: 10.25165/j.ijabe.20181101.3210.

SharathKumar, M. Malleshaiah, Ep Heuvelink e Leo F.M. Marcelis (ago. de 2020). «Vertical Farming: Moving from Genetic to Environmental Modification». Em: *Trends in Plant Science* 25.8, pp. 724–727. DOI: 10.1016/j.tplants.2020.05.012. URL: http://dx.doi.org/10.1016/j.tplants.2020.05.012.

Wong, Chui Eng et al. (2020). «Seeing the lights for leafy greens in indoor vertical farming». Em: *Trends in Food Science and Technology* 106, pp. 48–63. ISSN: 0924-2244. DOI: https://doi.org/10.1016/j.tifs.2020.09.031. URL: https://www.sciencedirect.com/science/article/pii/S092422442030621X.

Apêndice A

Restrições expandidas

De seguida, apresentamos as restrições associadas ao modelo apresentado na Secção 3.3. Para ganhar uma melhor familiaridade com os diversos índices das variáveis de decisão, apresentamos as várias restrições expandidas.

Procura: $\sum_{s \in T} x^6_{s,d-1,\tau} = p_d \quad \forall d \in D \cup \{\bar{d}\} = \{1, \ldots, 10\}$

$$
\begin{aligned}
d = 1: &\quad x^6_{t_1,0,\tau} + x^6_{t_2,0,\tau} + x^6_{t_3,0,\tau} + x^6_{t_4,0,\tau} = 0 \\
d = 2: &\quad x^6_{t_1,1,\tau} + x^6_{t_2,1,\tau} + x^6_{t_3,1,\tau} + x^6_{t_4,1,\tau} = 0 \\
d = 3: &\quad x^6_{t_1,2,\tau} + x^6_{t_2,2,\tau} + x^6_{t_3,2,\tau} + x^6_{t_4,2,\tau} = 0 \\
d = 4: &\quad x^6_{t_1,3,\tau} + x^6_{t_2,3,\tau} + x^6_{t_3,3,\tau} + x^6_{t_4,3,\tau} = 0 \\
d = 5: &\quad x^6_{t_1,4,\tau} + x^6_{t_2,4,\tau} + x^6_{t_3,4,\tau} + x^6_{t_4,4,\tau} = 0 \\
d = 6: &\quad x^6_{t_1,5,\tau} + x^6_{t_2,5,\tau} + x^6_{t_3,5,\tau} + x^6_{t_4,5,\tau} = 0 \\
d = 7: &\quad x^6_{t_1,6,\tau} + x^6_{t_2,6,\tau} + x^6_{t_3,6,\tau} + x^6_{t_4,6,\tau} = 1 \\
d = 8: &\quad x^6_{t_1,7,\tau} + x^6_{t_2,7,\tau} + x^6_{t_3,7,\tau} + x^6_{t_4,7,\tau} = 0 \\
d = 9: &\quad x^6_{t_1,8,\tau} + x^6_{t_2,8,\tau} + x^6_{t_3,8,\tau} + x^6_{t_4,8,\tau} = 0 \\
d = 10: &\quad x^6_{t_1,9,\tau} + x^6_{t_2,9,\tau} + x^6_{t_3,9,\tau} + x^6_{t_4,9,\tau} = 0
\end{aligned}
$$

Capacidade: $\sum_{r \in T'} \sum_{\substack{g \in \{1,\ldots,6\} \\ k_g = k}} x^g_{s,d,r} \leq 5 \cdot y_{s,\,d,\,k} \quad \forall s \in T\,, \forall k \in K\,, \forall d \in \{1, 2, \ldots, 9\}$

Para este conjunto de restrições, indicamos apenas quatro, que estão associadas aos

parâmetros $d = 1$, prateleira t_1, e configurações $\{I_1, I_2, I_3, I_4\}$. As restantes restrições deste conjunto são omitidas.

$$\begin{aligned}
\mathbf{I_1}: & \; x^1_{t_1,1,t_1} + x^1_{t_1,1,t_2} + x^1_{t_1,1,t_3} + x^1_{t_1,1,t_4} + x^1_{t_1,1,\tau} \leq 5 \cdot \mathbf{y_{t_1,1,I_1}} \\
\mathbf{I_2}: & \; x^2_{t_1,1,t_1} + x^2_{t_1,1,t_2} + x^2_{t_1,1,t_3} + x^2_{t_1,1,t_4} + x^2_{t_1,1,\tau} \\
& + x^3_{t_1,1,t_1} + x^3_{t_1,1,t_2} + x^3_{t_1,1,t_3} + x^3_{t_1,1,t_4} + x^3_{t_1,1,\tau} \leq 5 \cdot \mathbf{y_{t_1,1,I_2}} \\
\mathbf{I_3}: & \; x^4_{t_1,1,t_1} + x^4_{t_1,1,t_2} + x^4_{t_1,1,t_3} + x^4_{t_1,1,t_4} + x^4_{t_1,1,\tau} \\
& + x^5_{t_1,1,t_1} + x^5_{t_1,1,t_2} + x^5_{t_1,1,t_3} + x^5_{t_1,1,t_4} + x^5_{t_1,1,\tau} \leq 5 \cdot \mathbf{y_{t_1,1,I_3}} \\
\mathbf{I_4}: & \; x^6_{t_1,1,t_1} + x^6_{t_1,1,t_2} + x^6_{t_1,1,t_3} + x^6_{t_1,1,t_4} + x^6_{t_1,1,\tau} \leq 5 \cdot \mathbf{y_{t_1,1,I_4}}
\end{aligned}$$

Plantação: $\sum_{s \in T} x^0_{\sigma,d-6-1,s} = p_d \quad \forall d \in \{7, \ldots, 10\}$

$$\begin{aligned}
d = 7: & \quad x^0_{\sigma,0,t_1} + x^0_{\sigma,0,t_2} + x^0_{\sigma,0,t_3} + x^0_{\sigma,0,t_4} = 1 \\
d = 8: & \quad x^0_{\sigma,1,t_1} + x^0_{\sigma,1,t_2} + x^0_{\sigma,1,t_3} + x^0_{\sigma,1,t_4} = 0 \\
d = 9: & \quad x^0_{\sigma,2,t_1} + x^0_{\sigma,2,t_2} + x^0_{\sigma,2,t_3} + x^0_{\sigma,2,t_4} = 0 \\
d = 10: & \quad x^0_{\sigma,3,t_1} + x^0_{\sigma,3,t_2} + x^0_{\sigma,3,t_3} + x^0_{\sigma,3,t_4} = 0
\end{aligned}$$

Configuração: $\sum_{k \in K} y_{s,d,k} \leq 1 \quad \forall s \in T, \forall d \in D = \{1, \ldots, 9\}.$

$$\begin{aligned}
s = t_1: \quad d = 1: & \quad y_{t_1,1,I_1} + y_{t_1,1,I_2} + y_{t_1,1,I_3} + y_{t_1,1,I_4} \leq 1 \\
d = 2: & \quad y_{t_1,2,I_1} + y_{t_1,2,I_2} + y_{t_1,2,I_3} + y_{t_1,2,I_4} \leq 1 \\
d = 3: & \quad y_{t_1,3,I_1} + y_{t_1,3,I_2} + y_{t_1,3,I_3} + y_{t_1,3,I_4} \leq 1 \\
d = 4: & \quad y_{t_1,4,I_1} + y_{t_1,4,I_2} + y_{t_1,4,I_3} + y_{t_1,4,I_4} \leq 1 \\
d = 5: & \quad y_{t_1,5,I_1} + y_{t_1,5,I_2} + y_{t_1,5,I_3} + y_{t_1,5,I_4} \leq 1 \\
d = 6: & \quad y_{t_1,6,I_1} + y_{t_1,6,I_2} + y_{t_1,6,I_3} + y_{t_1,6,I_4} \leq 1 \\
d = 7: & \quad y_{t_1,7,I_1} + y_{t_1,7,I_2} + y_{t_1,7,I_3} + y_{t_1,7,I_4} \leq 1 \\
d = 8: & \quad y_{t_1,8,I_1} + y_{t_1,8,I_2} + y_{t_1,8,I_3} + y_{t_1,8,I_4} \leq 1 \\
d = 9: & \quad y_{t_1,9,I_1} + y_{t_1,9,I_2} + y_{t_1,9,I_3} + y_{t_1,9,I_4} \leq 1 \\
s = t_2: \quad d = 1: & \quad y_{t_2,1,I_1} + y_{t_2,1,I_2} + y_{t_2,1,I_3} + y_{t_2,1,I_4} \leq 1 \\
d = 2: & \quad y_{t_2,2,I_1} + y_{t_2,2,I_2} + y_{t_2,2,I_3} + y_{t_2,2,I_4} \leq 1
\end{aligned}$$

$$
\begin{aligned}
& & d=3: \quad & y_{t_2,3,I_1}+y_{t_2,3,I_2}+y_{t_2,3,I_3}+y_{t_2,3,I_4}\leq 1\\
& & d=4: \quad & y_{t_2,4,I_1}+y_{t_2,4,I_2}+y_{t_2,4,I_3}+y_{t_2,4,I_4}\leq 1\\
& & d=5: \quad & y_{t_2,5,I_1}+y_{t_2,5,I_2}+y_{t_2,5,I_3}+y_{t_2,5,I_4}\leq 1\\
& & d=6: \quad & y_{t_2,6,I_1}+y_{t_2,6,I_2}+y_{t_2,6,I_3}+y_{t_2,6,I_4}\leq 1\\
& & d=7: \quad & y_{t_2,7,I_1}+y_{t_2,7,I_2}+y_{t_2,7,I_3}+y_{t_2,7,I_4}\leq 1\\
& & d=8: \quad & y_{t_2,8,I_1}+y_{t_2,8,I_2}+y_{t_2,8,I_3}+y_{t_2,8,I_4}\leq 1\\
& & d=9: \quad & y_{t_2,9,I_1}+y_{t_2,9,I_2}+y_{t_2,9,I_3}+y_{t_2,9,I_4}\leq 1\\
s=t_3: \quad & d=1: \quad & y_{t_3,1,I_1}+y_{t_3,1,I_2}+y_{t_3,1,I_3}+y_{t_3,1,I_4}\leq 1\\
& & d=2: \quad & y_{t_3,2,I_1}+y_{t_3,2,I_2}+y_{t_3,2,I_3}+y_{t_3,2,I_4}\leq 1\\
& & d=3: \quad & y_{t_3,3,I_1}+y_{t_3,3,I_2}+y_{t_3,3,I_3}+y_{t_3,3,I_4}\leq 1\\
& & d=4: \quad & y_{t_3,4,I_1}+y_{t_3,4,I_2}+y_{t_3,4,I_3}+y_{t_3,4,I_4}\leq 1\\
& & d=5: \quad & y_{t_3,5,I_1}+y_{t_3,5,I_2}+y_{t_3,5,I_3}+y_{t_3,5,I_4}\leq 1\\
& & d=6: \quad & y_{t_3,6,I_1}+y_{t_3,6,I_2}+y_{t_3,6,I_3}+y_{t_3,6,I_4}\leq 1\\
& & d=7: \quad & y_{t_3,7,I_1}+y_{t_3,7,I_2}+y_{t_3,7,I_3}+y_{t_3,7,I_4}\leq 1\\
& & d=8: \quad & y_{t_3,8,I_1}+y_{t_3,8,I_2}+y_{t_3,8,I_3}+y_{t_3,8,I_4}\leq 1\\
& & d=9: \quad & y_{t_3,9,I_1}+y_{t_3,9,I_2}+y_{t_3,9,I_3}+y_{t_3,9,I_4}\leq 1\\
s=t_4: \quad & d=1: \quad & y_{t_4,1,I_1}+y_{t_4,1,I_2}+y_{t_4,1,I_3}+y_{t_4,1,I_4}\leq 1\\
& & d=2: \quad & y_{t_4,2,I_1}+y_{t_4,2,I_2}+y_{t_4,2,I_3}+y_{t_4,2,I_4}\leq 1\\
& & d=3: \quad & y_{t_4,3,I_1}+y_{t_4,3,I_2}+y_{t_4,3,I_3}+y_{t_4,3,I_4}\leq 1\\
& & d=4: \quad & y_{t_4,4,I_1}+y_{t_4,4,I_2}+y_{t_4,4,I_3}+y_{t_4,4,I_4}\leq 1\\
& & d=5: \quad & y_{t_4,5,I_1}+y_{t_4,5,I_2}+y_{t_4,5,I_3}+y_{t_4,5,I_4}\leq 1\\
& & d=6: \quad & y_{t_4,6,I_1}+y_{t_4,6,I_2}+y_{t_4,6,I_3}+y_{t_4,6,I_4}\leq 1\\
& & d=7: \quad & y_{t_4,7,I_1}+y_{t_4,7,I_2}+y_{t_4,7,I_3}+y_{t_4,7,I_4}\leq 1\\
& & d=8: \quad & y_{t_4,8,I_1}+y_{t_4,8,I_2}+y_{t_4,8,I_3}+y_{t_4,8,I_4}\leq 1\\
& & d=9: \quad & y_{t_4,9,I_1}+y_{t_4,9,I_2}+y_{t_4,9,I_3}+y_{t_4,9,I_4}\leq 1
\end{aligned}
$$

Equilíbrio: $\sum_{r\in T'} x^{g}_{r,d-1,s}=\sum_{r\in T'} x^{g+1}_{s,d,r} \quad \forall g\in\{0,\ldots,6\},\forall s\in T,\forall d\in D=\{1,\ldots,9\}$

Para este conjunto de restrições, indicamos apenas as que estão associadas aos parâ-

metros $d \in D$, prateleiras $s \in T$, e $g =, 0$ omitindo as restantes restrições.

$$
\begin{aligned}
d = 1: \quad & s = t_1: && x^0_{\sigma,0,t_1} + x^0_{t_1,0,t_1} = x^1_{t_1,1,t_1} + x^1_{t_1,1,t_2} + x^1_{t_1,1,t_3} + x^1_{t_1,1,t_4} + x^1_{t_1,1,\tau} \\
& s = t_2: && x^0_{\sigma,0,t_2} + x^0_{t_1,0,t_2} + x^0_{t_2,0,t_2} = x^1_{t_2,1,t_2} + x^1_{t_2,1,t_3} + x^1_{t_2,1,t_4} + x^1_{t_2,1,\tau} \\
& s = t_3: && x^0_{\sigma,0,t_3} + x^0_{t_1,0,t_3} + x^0_{t_2,0,t_3} + x^0_{t_3,0,t_3} = x^1_{t_3,1,t_3} + x^1_{t_3,1,t_4} + x^1_{t_3,1,\tau} \\
& s = t_4: && x^0_{\sigma,0,t_4} + x^0_{t_1,0,t_4} + x^0_{t_2,0,t_4} + x^0_{t_3,0,t_4} + x^0_{t_4,0,t_4} = x^1_{t_4,1,t_4} + x^1_{t_4,1,\tau} \\
d = 2: \quad & s = t_1: && x^0_{\sigma,1,t_1} + x^0_{t_1,1,t_1} = x^1_{t_1,2,t_1} + x^1_{t_1,2,t_2} + x^1_{t_1,2,t_3} + x^1_{t_1,2,t_4} + x^1_{t_1,2,\tau} \\
& s = t_2: && x^0_{\sigma,1,t_2} + x^0_{t_1,1,t_2} + x^0_{t_2,1,t_2} = x^1_{t_2,2,t_2} + x^1_{t_2,2,t_3} + x^1_{t_2,2,t_4} + x^1_{t_2,2,\tau} \\
& s = t_3: && x^0_{\sigma,1,t_3} + x^0_{t_1,1,t_3} + x^0_{t_2,1,t_3} + x^0_{t_3,1,t_3} = x^1_{t_3,2,t_3} + x^1_{t_3,2,t_4} + x^1_{t_3,2,\tau} \\
& s = t_4: && x^0_{\sigma,1,t_4} + x^0_{t_1,1,t_4} + x^0_{t_2,1,t_4} + x^0_{t_3,1,t_4} + x^0_{t_4,1,t_4} = x^1_{t_4,2,t_4} + x^1_{t_4,2,\tau} \\
d = 3: \quad & s = t_1: && x^0_{\sigma,2,t_1} + x^0_{t_1,2,t_1} = x^1_{t_1,3,t_1} + x^1_{t_1,3,t_2} + x^1_{t_1,3,t_3} + x^1_{t_1,3,t_4} + x^1_{t_1,3,\tau} \\
& s = t_2: && x^0_{\sigma,2,t_2} + x^0_{t_1,2,t_2} + x^0_{t_2,2,t_2} = x^1_{t_2,3,t_2} + x^1_{t_2,3,t_3} + x^1_{t_2,3,t_4} + x^1_{t_2,3,\tau} \\
& s = t_3: && x^0_{\sigma,2,t_3} + x^0_{t_1,2,t_3} + x^0_{t_2,2,t_3} + x^0_{t_3,2,t_3} = x^1_{t_3,3,t_3} + x^1_{t_3,3,t_4} + x^1_{t_3,3,\tau} \\
& s = t_4: && x^0_{\sigma,2,t_4} + x^0_{t_1,2,t_4} + x^0_{t_2,2,t_4} + x^0_{t_3,2,t_4} + x^0_{t_4,2,t_4} = x^1_{t_4,3,t_4} + x^1_{t_4,3,\tau} \\
d = 4: \quad & s = t_1: && x^0_{\sigma,3,t_1} + x^0_{t_1,3,t_1} = x^1_{t_1,4,t_1} + x^1_{t_1,4,t_2} + x^1_{t_1,4,t_3} + x^1_{t_1,4,t_4} + x^1_{t_1,4,\tau} \\
& s = t_2: && x^0_{\sigma,3,t_2} + x^0_{t_1,3,t_2} + x^0_{t_2,3,t_2} = x^1_{t_2,4,t_2} + x^1_{t_2,4,t_3} + x^1_{t_2,4,t_4} + x^1_{t_2,4,\tau} \\
& s = t_3: && x^0_{\sigma,3,t_3} + x^0_{t_1,3,t_3} + x^0_{t_2,3,t_3} + x^0_{t_3,3,t_3} = x^1_{t_3,4,t_3} + x^1_{t_3,4,t_4} + x^1_{t_3,4,\tau} \\
& s = t_4: && x^0_{\sigma,3,t_4} + x^0_{t_1,3,t_4} + x^0_{t_2,3,t_4} + x^0_{t_3,3,t_4} + x^0_{t_4,3,t_4} = x^1_{t_4,4,t_4} + x^1_{t_4,4,\tau} \\
d = 5: \quad & s = t_1: && x^0_{\sigma,4,t_1} + x^0_{t_1,4,t_1} = x^1_{t_1,5,t_1} + x^1_{t_1,5,t_2} + x^1_{t_1,5,t_3} + x^1_{t_1,5,t_4} + x^1_{t_1,5,\tau} \\
& s = t_2: && x^0_{\sigma,4,t_2} + x^0_{t_1,4,t_2} + x^0_{t_2,4,t_2} = x^1_{t_2,5,t_2} + x^1_{t_2,5,t_3} + x^1_{t_2,5,t_4} + x^1_{t_2,5,\tau} \\
& s = t_3: && x^0_{\sigma,4,t_3} + x^0_{t_1,4,t_3} + x^0_{t_2,4,t_3} + x^0_{t_3,4,t_3} = x^1_{t_3,5,t_3} + x^1_{t_3,5,t_4} + x^1_{t_3,5,\tau} \\
& s = t_4: && x^0_{\sigma,4,t_4} + x^0_{t_1,4,t_4} + x^0_{t_2,4,t_4} + x^0_{t_3,4,t_4} + x^0_{t_4,4,t_4} = x^1_{t_4,5,t_4} + x^1_{t_4,5,\tau} \\
d = 6: \quad & s = t_1: && x^0_{\sigma,5,t_1} + x^0_{t_1,5,t_1} = x^1_{t_1,6,t_1} + x^1_{t_1,6,t_2} + x^1_{t_1,6,t_3} + x^1_{t_1,6,t_4} + x^1_{t_1,6,\tau} \\
& s = t_2: && x^0_{\sigma,5,t_2} + x^0_{t_1,5,t_2} + x^0_{t_2,5,t_2} = x^1_{t_2,6,t_2} + x^1_{t_2,6,t_3} + x^1_{t_2,6,t_4} + x^1_{t_2,6,\tau} \\
& s = t_3: && x^0_{\sigma,5,t_3} + x^0_{t_1,5,t_3} + x^0_{t_2,5,t_3} + x^0_{t_3,5,t_3} = x^1_{t_3,6,t_3} + x^1_{t_3,6,t_4} + x^1_{t_3,6,\tau} \\
& s = t_4: && x^0_{\sigma,5,t_4} + x^0_{t_1,5,t_4} + x^0_{t_2,5,t_4} + x^0_{t_3,5,t_4} + x^0_{t_4,5,t_4} = x^1_{t_4,6,t_4} + x^1_{t_4,6,\tau} \\
d = 7: \quad & s = t_1: && x^0_{\sigma,6,t_1} + x^0_{t_1,6,t_1} = x^1_{t_1,7,t_1} + x^1_{t_1,7,t_2} + x^1_{t_1,7,t_3} + x^1_{t_1,7,t_4} + x^1_{t_1,7,\tau} \\
& s = t_2: && x^0_{\sigma,6,t_2} + x^0_{t_1,6,t_2} + x^0_{t_2,6,t_2} = x^1_{t_2,7,t_2} + x^1_{t_2,7,t_3} + x^1_{t_2,7,t_4} + x^1_{t_2,7,\tau} \\
& s = t_3: && x^0_{\sigma,6,t_3} + x^0_{t_1,6,t_3} + x^0_{t_2,6,t_3} + x^0_{t_3,6,t_3} = x^1_{t_3,7,t_3} + x^1_{t_3,7,t_4} + x^1_{t_3,7,\tau}
\end{aligned}
$$

$$
\begin{aligned}
& & s = t_4: \quad & x^0_{\sigma,6,t_4} + x^0_{t_1,6,t_4} + x^0_{t_2,6,t_4} + x^0_{t_3,6,t_4} + x^0_{t_4,6,t_4} = x^1_{t_4,7,t_4} + x^1_{t_4,7,\tau} \\
d = 8: & \quad s = t_1: \quad & x^0_{\sigma,7,t_1} + x^0_{t_1,7,t_1} = x^1_{t_1,8,t_1} + x^1_{t_1,8,t_2} + x^1_{t_1,8,t_3} + x^1_{t_1,8,t_4} + x^1_{t_1,8,\tau} \\
& \quad s = t_2: \quad & x^0_{\sigma,7,t_2} + x^0_{t_1,7,t_2} + x^0_{t_2,7,t_2} = x^1_{t_2,8,t_2} + x^1_{t_2,8,t_3} + x^1_{t_2,8,t_4} + x^1_{t_2,8,\tau} \\
& \quad s = t_3: \quad & x^0_{\sigma,7,t_3} + x^0_{t_1,7,t_3} + x^0_{t_2,7,t_3} + x^0_{t_3,7,t_3} = x^1_{t_3,8,t_3} + x^1_{t_3,8,t_4} + x^1_{t_3,8,\tau} \\
& \quad s = t_4: \quad & x^0_{\sigma,7,t_4} + x^0_{t_1,7,t_4} + x^0_{t_2,7,t_4} + x^0_{t_3,7,t_4} + x^0_{t_4,7,t_4} = x^1_{t_4,8,t_4} + x^1_{t_4,8,\tau} \\
d = 9: & \quad s = t_1: \quad & x^0_{\sigma,8,t_1} + x^0_{t_1,8,t_1} = x^1_{t_1,9,t_1} + x^1_{t_1,9,t_2} + x^1_{t_1,9,t_3} + x^1_{t_1,9,t_4} + x^1_{t_1,9,\tau} \\
& \quad s = t_2: \quad & x^0_{\sigma,8,t_2} + x^0_{t_1,8,t_2} + x^0_{t_2,8,t_2} = x^1_{t_2,9,t_2} + x^1_{t_2,9,t_3} + x^1_{t_2,9,t_4} + x^1_{t_2,9,\tau} \\
& \quad s = t_3: \quad & x^0_{\sigma,8,t_3} + x^0_{t_1,8,t_3} + x^0_{t_2,8,t_3} + x^0_{t_3,8,t_3} = x^1_{t_3,9,t_3} + x^1_{t_3,9,t_4} + x^1_{t_3,9,\tau} \\
& \quad s = t_4: \quad & x^0_{\sigma,8,t_4} + x^0_{t_1,8,t_4} + x^0_{t_2,8,t_4} + x^0_{t_3,8,t_4} + x^0_{t_4,8,t_4} = x^1_{t_4,9,t_4} + x^1_{t_4,9,\tau}
\end{aligned}
$$

Fixação de variáveis:

FV1: $\forall d \in \{0, 1, \ldots, 9\}$

$$
\begin{aligned}
x^0_{t_1,d,\sigma} &= x^0_{t_2,d,\sigma} = x^0_{t_3,d,\sigma} = x^0_{t_4,d,\sigma} = 0 \\
x^1_{t_1,d,\sigma} &= x^1_{t_2,d,\sigma} = x^1_{t_3,d,\sigma} = x^1_{t_4,d,\sigma} = 0 \\
x^2_{t_1,d,\sigma} &= x^2_{t_2,d,\sigma} = x^2_{t_3,d,\sigma} = x^2_{t_4,d,\sigma} = 0 \\
x^3_{t_1,d,\sigma} &= x^3_{t_2,d,\sigma} = x^3_{t_3,d,\sigma} = x^3_{t_4,d,\sigma} = 0 \\
x^4_{t_1,d,\sigma} &= x^4_{t_2,d,\sigma} = x^4_{t_3,d,\sigma} = x^4_{t_4,d,\sigma} = 0 \\
x^5_{t_1,d,\sigma} &= x^5_{t_2,d,\sigma} = x^5_{t_3,d,\sigma} = x^5_{t_4,d,\sigma} = 0 \\
x^6_{t_1,d,\sigma} &= x^6_{t_2,d,\sigma} = x^6_{t_3,d,\sigma} = x^6_{t_4,d,\sigma} = 0
\end{aligned}
$$

$$
\begin{aligned}
x^0_{\tau,d,t_1} &= x^0_{\tau,d,t_2} = x^0_{\tau,d,t_3} = x^0_{\tau,d,t_4} = 0 \\
x^1_{\tau,d,t_1} &= x^1_{\tau,d,t_2} = x^1_{\tau,d,t_3} = x^1_{\tau,d,t_4} = 0 \\
x^2_{\tau,d,t_1} &= x^2_{\tau,d,t_2} = x^2_{\tau,d,t_3} = x^2_{\tau,d,t_4} = 0 \\
x^3_{\tau,d,t_1} &= x^3_{\tau,d,t_2} = x^3_{\tau,d,t_3} = x^3_{\tau,d,t_4} = 0 \\
x^4_{\tau,d,t_1} &= x^4_{\tau,d,t_2} = x^4_{\tau,d,t_3} = x^4_{\tau,d,t_4} = 0 \\
x^5_{\tau,d,t_1} &= x^5_{\tau,d,t_2} = x^5_{\tau,d,t_3} = x^5_{\tau,d,t_4} = 0 \\
x^6_{\tau,d,t_1} &= x^6_{\tau,d,t_2} = x^6_{\tau,d,t_3} = x^6_{\tau,d,t_4} = 0 \\
x^0_{\sigma,d,\tau} &= x^1_{\sigma,d,\tau} = x^2_{\sigma,d,\tau} = x^3_{\sigma,d,\tau} = x^4_{\sigma,d,\tau} = x^5_{\sigma,d,\tau} = x^6_{\sigma,d,\tau} = 0
\end{aligned}
$$

FV2: $\forall d \in \{0, 1, \ldots, 9\}$

$$x^1_{\sigma,d,\sigma} = x^1_{\sigma,d,t_1} = x^1_{\sigma,d,t_2} = x^1_{\sigma,d,t_3} = x^1_{\sigma,d,t_4} = x^1_{\sigma,d,\tau} = 0$$
$$x^2_{\sigma,d,\sigma} = x^2_{\sigma,d,t_1} = x^2_{\sigma,d,t_2} = x^2_{\sigma,d,t_3} = x^2_{\sigma,d,t_4} = x^2_{\sigma,d,\tau} = 0$$
$$x^3_{\sigma,d,\sigma} = x^3_{\sigma,d,t_1} = x^3_{\sigma,d,t_2} = x^3_{\sigma,d,t_3} = x^3_{\sigma,d,t_4} = x^3_{\sigma,d,\tau} = 0$$
$$x^4_{\sigma,d,\sigma} = x^4_{\sigma,d,t_1} = x^4_{\sigma,d,t_2} = x^4_{\sigma,d,t_3} = x^4_{\sigma,d,t_4} = x^4_{\sigma,d,\tau} = 0$$
$$x^5_{\sigma,d,\sigma} = x^5_{\sigma,d,t_1} = x^5_{\sigma,d,t_2} = x^5_{\sigma,d,t_3} = x^5_{\sigma,d,t_4} = x^5_{\sigma,d,\tau} = 0$$
$$x^6_{\sigma,d,\sigma} = x^6_{\sigma,d,t_1} = x^6_{\sigma,d,t_2} = x^6_{\sigma,d,t_3} = x^6_{\sigma,d,t_4} = x^6_{\sigma,d,\tau} = 0$$

$$x^0_{\sigma,d,\tau} = x^0_{t_1,d,\tau} = x^0_{t_2,d,\tau} = x^0_{t_3,d,\tau} = x^0_{t_4,d,\tau} = x^0_{\tau,d,\tau} = 0$$
$$x^1_{\sigma,d,\tau} = x^1_{t_1,d,\tau} = x^1_{t_2,d,\tau} = x^1_{t_3,d,\tau} = x^1_{t_4,d,\tau} = x^1_{\tau,d,\tau} = 0$$
$$x^2_{\sigma,d,\tau} = x^2_{t_1,d,\tau} = x^2_{t_2,d,\tau} = x^2_{t_3,d,\tau} = x^2_{t_4,d,\tau} = x^2_{\tau,d,\tau} = 0$$
$$x^3_{\sigma,d,\tau} = x^3_{t_1,d,\tau} = x^3_{t_2,d,\tau} = x^3_{t_3,d,\tau} = x^3_{t_4,d,\tau} = x^3_{\tau,d,\tau} = 0$$
$$x^4_{\sigma,d,\tau} = x^4_{t_1,d,\tau} = x^4_{t_2,d,\tau} = x^4_{t_3,d,\tau} = x^4_{t_4,d,\tau} = x^4_{\tau,d,\tau} = 0$$
$$x^5_{\sigma,d,\tau} = x^5_{t_1,d,\tau} = x^5_{t_2,d,\tau} = x^5_{t_3,d,\tau} = x^5_{t_4,d,\tau} = x^5_{\tau,d,\tau} = 0$$

$$x^0_{t_1,d,t_1} = x^0_{t_1,d,t_2} = x^0_{t_1,d,t_3} = x^0_{t_1,d,t_4} = 0$$
$$x^0_{t_2,d,t_1} = x^0_{t_2,d,t_2} = x^0_{t_2,d,t_3} = x^0_{t_2,d,t_4} = 0$$
$$x^0_{t_3,d,t_1} = x^0_{t_3,d,t_2} = x^0_{t_3,d,t_3} = x^0_{t_3,d,t_4} = 0$$
$$x^0_{t_4,d,t_1} = x^0_{t_4,d,t_2} = x^0_{t_4,d,t_3} = x^0_{t_4,d,t_4} = 0$$
$$x^6_{t_1,d,t_1} = x^6_{t_1,d,t_2} = x^6_{t_1,d,t_3} = x^6_{t_1,d,t_4} = 0$$
$$x^6_{t_2,d,t_1} = x^6_{t_2,d,t_2} = x^6_{t_2,d,t_3} = x^6_{t_2,d,t_4} = 0$$
$$x^6_{t_3,d,t_1} = x^6_{t_3,d,t_2} = x^6_{t_3,d,t_3} = x^6_{t_3,d,t_4} = 0$$
$$x^6_{t_4,d,t_1} = x^6_{t_4,d,t_2} = x^6_{t_4,d,t_3} = x^6_{t_4,d,t_4} = 0$$

FV3: $\forall d \in \{0, 1, \ldots, 9\}$

$$x^0_{t_1,d,s} = x^0_{t_2,d,s} = x^0_{t_3,d,s} = x^0_{t_4,d,s} = x^0_{\tau,d,\sigma} = 0 \quad \forall s \in T'$$
$$x^6_{s,d,\sigma} = x^6_{s,d,t_1} = x^6_{s,d,t_2} = x^6_{s,d,t_3} = x^6_{s,d,t_4} = 0 \quad \forall s \in T'$$

FV4: $d' = \max_{d\in\{1,\dots,9\}}\{d : p_d > 0\}$ o último dia com procura. Temos $d' = 7$.

$$x^g_{s,d,r} = 0 \quad \forall s, r \in T,\ \forall d \in \{d' - 6 + 1, \dots, d'\}, \forall g \in \{0, \dots,\ 6 - (d' - d)\},\ \text{ou seja,}$$
$$x^g_{r,d,s} = 0 \quad \forall r, s \in T,\ \forall d \in \{2, 3, 4, 5, 6, 7\}, \forall g \in \{0, \dots,\ d - 1\}$$

Então:

$$\begin{aligned}
\forall r, s \in T: \quad & d = 2: && x^0_{r,2,s} = x^1_{r,2,s} = 0\\
& d = 3: && x^0_{r,3,s} = x^1_{r,3,s} = x^2_{r,3,s} = 0\\
& d = 4: && x^0_{r,4,s} = x^1_{r,4,s} = x^2_{r,4,s} = x^3_{r,4,s} = 0\\
& d = 5: && x^0_{r,5,s} = x^1_{r,5,s} = x^2_{r,5,s} = x^3_{r,5,s} = x^4_{r,5,s} = 0\\
& d = 6: && x^0_{r,6,s} = x^1_{r,6,s} = x^2_{r,6,s} = x^3_{r,6,s} = x^4_{r,6,s} = x^5_{r,6,s} = 0\\
& d = 7: && x^0_{r,7,s} = x^1_{r,7,s} = x^2_{r,7,s} = x^3_{r,7,s} = x^4_{r,7,s} = x^5_{r,7,s} = x^6_{r,7,s} = 0
\end{aligned}$$

FV5: $x^g_{r,d,s} = 0 \quad \forall r, s \in T', \forall d \in D' : d < 6, \forall g \in \{d + 1, \dots, 6\}$.

$$\begin{aligned}
\forall r, s \in T': \quad & d = 0: && x^1_{r,0,s} = x^2_{r,0,s} = x^3_{r,0,s} = x^4_{r,0,s} = x^5_{r,0,s} = x^6_{r,0,s} = 0\\
& d = 1: && x^2_{r,1,s} = x^3_{r,1,s} = x^4_{r,1,s} = x^5_{r,1,s} = x^6_{r,1,s} = 0\\
& d = 2: && x^3_{r,2,s} = x^4_{r,2,s} = x^5_{r,2,s} = x^6_{r,2,s} = 0\\
& d = 3: && x^4_{r,3,s} = x^5_{r,3,s} = x^6_{r,3,s} = 0\\
& d = 4: && x^5_{r,4,s} = x^6_{r,4,s} = 0\\
& d = 5: && x^6_{r,5,s} = 0
\end{aligned}$$

FV6: $\forall\, r \in T$:

- $x^0_{\sigma,d-6-1,r} \le p_d \quad \forall\, d \in \{6 + 1, \dots, 10\}$

$$\begin{aligned}
d = 7: \quad & x^0_{\sigma,0,t_1} \le 1 && d = 8: \quad x^0_{\sigma,1,t_1} \le 0 && d = 9: \quad x^0_{\sigma,2,t_1} \le 0 && d = 10: \quad x^0_{\sigma,3,t_1} \le 0\\
& x^0_{\sigma,0,t_2} \le 1 && x^0_{\sigma,1,t_2} \le 0 && x^0_{\sigma,2,t_2} \le 0 && x^0_{\sigma,3,t_2} \le 0\\
& x^0_{\sigma,0,t_3} \le 1 && x^0_{\sigma,1,t_3} \le 0 && x^0_{\sigma,2,t_3} \le 0 && x^0_{\sigma,3,t_3} \le 0\\
& x^0_{\sigma,0,t_4} \le 1 && x^0_{\sigma,1,t_4} \le 0 && x^0_{\sigma,2,t_4} \le 0 && x^0_{\sigma,3,t_4} \le 0
\end{aligned}$$

- $x^6_{r,d-1,\tau} \leq p_d \quad \forall d \in \{1, 2, \ldots, 10\}$

$$
\begin{aligned}
d = 1: \quad & x^6_{t_1,0,\tau} \leq 0 & d = 2: \quad & x^6_{t_1,1,\tau} \leq 0 \\
& x^6_{t_2,0,\tau} \leq 0 & & x^6_{t_2,1,\tau} \leq 0 \\
& x^6_{t_3,0,\tau} \leq 0 & & x^6_{t_3,1,\tau} \leq 0 \\
& x^6_{t_4,0,\tau} \leq 0 & & x^6_{t_4,1,\tau} \leq 0 \\
d = 3: \quad & x^6_{t_1,2,\tau} \leq 0 & d = 4: \quad & x^6_{t_1,3,\tau} \leq 0 \\
& x^6_{t_2,2,\tau} \leq 0 & & x^6_{t_2,3,\tau} \leq 0 \\
& x^6_{t_3,2,\tau} \leq 0 & & x^6_{t_3,3,\tau} \leq 0 \\
& x^6_{t_4,2,\tau} \leq 0 & & x^6_{t_4,3,\tau} \leq 0 \\
d = 5: \quad & x^6_{t_1,4,\tau} \leq 0 & d = 6: \quad & x^6_{t_1,5,\tau} \leq 0 \\
& x^6_{t_2,4,\tau} \leq 0 & & x^6_{t_2,5,\tau} \leq 0 \\
& x^6_{t_3,4,\tau} \leq 0 & & x^6_{t_3,5,\tau} \leq 0 \\
& x^6_{t_4,4,\tau} \leq 0 & & x^6_{t_4,5,\tau} \leq 0 \\
d = 7: \quad & x^6_{t_1,6,\tau} \leq 1 & d = 8: \quad & x^6_{t_1,7,\tau} \leq 0 \\
& x^6_{t_2,6,\tau} \leq 1 & & x^6_{t_2,7,\tau} \leq 0 \\
& x^6_{t_3,6,\tau} \leq 1 & & x^6_{t_3,7,\tau} \leq 0 \\
& x^6_{t_4,6,\tau} \leq 1 & & x^6_{t_4,7,\tau} \leq 0 \\
d = 9: \quad & x^6_{t_1,8,\tau} \leq 0 & d = 10: \quad & x^6_{t_1,9,\tau} \leq 0 \\
& x^6_{t_2,8,\tau} \leq 0 & & x^6_{t_2,9,\tau} \leq 0 \\
& x^6_{t_3,8,\tau} \leq 0 & & x^6_{t_3,9,\tau} \leq 0 \\
& x^6_{t_4,8,\tau} \leq 0 & & x^6_{t_4,9,\tau} \leq 0
\end{aligned}
$$

FV7: $\eta_{d,k} \in \mathbb{N}$, o número de paletes que requerem a configuração k no dia d, temos:

$$\eta_{d,k} = \sum_{d'=d+1}^{d+6} \sum_{k_{6-(d'-d-1)}=k} p_{d'} \quad \forall d \in \{1, \ldots, 9\}, \forall k \in K.$$

Então $\quad y_{s,d,k} = 0 \quad \forall s \in T, \forall d \in D, \forall k \in K : \eta_{d,k} = 0.$

Para o nosso caso, verifica-se que $p_{d'} = 1$ se $d' = 7$, e $p_{d'} = 0$ se $d' \in \{1, \ldots, 6, 8, 9\}$.

Cálculo de $\eta_{d,\,k}$: $\eta_{d,\,k} = \sum_{d'=d+1}^{d+6} \sum_{k_{7-(d'-d)}=k} p_{d'} \quad \forall d \in \{1, \ldots, 9\}, \forall k \in K.$

$$\eta_{1,I_1} = \sum_{d'=1+1}^{1+6} \sum_{k_{7-(d'-1)}=I_1} p_{d'} = \sum_{d'=2}^{7} \sum_{k_{8-d'}=I_1} p_{d'} = p_7 = 1$$

Como $p_{d'} = 0, \forall d' \in \{1, \ldots, 6, 8, 9\}$, a única parcela não nula corresponde a $d' = 7$. Nesse caso, $k_{8-7} = k_1 = I_1$, logo $\eta_{1,I_1} = 1$. Além disso, como $I_1 \neq I_2, I_3, I_4$, conclui-se que $\eta_{1,I_2} = \eta_{1,I_3} = \eta_{1,I_4} = 0$.

Para $d = 2$ temos:

$$\eta_{2,k} = \sum_{d'=2+1}^{2+6} \sum_{k_{7-(d'-2)}=k} p_{d'} = \sum_{d'=3}^{8} \sum_{k_{9-d'}=k} p_{d'}.$$

Como $p_{d'} \neq 0$ se $d' = 7$, conclui-se que $\eta_{2,k} = 0$ se $k_{9-d'} \neq k$ com $d' = 7$, isto é, $k_{9-7} = k_2 \neq k$. Ora, como $k_2 = I_2$, temos $\eta_{2,I_1} = \eta_{2,I_3} = \eta_{2,I_4} = 0$ e $\eta_{2,I_2} = 1$.

Para $d = 3$ temos:

$$\eta_{3,k} = \sum_{d'=3+1}^{3+6} \sum_{k_{7-(d'-3)}=k} p_{d'} = \sum_{d'=4}^{9} \sum_{k_{10-d'}=k} p_{d'}.$$

Tal como no caso anterior, $\eta_{3,k} = 0$ se $k_{10-d'} \neq k$ onde $d' = 7$. Como $k_{10-d'} = k_3 = I_2$, concluímos que $\eta_{3,I_1} = \eta_{3,I_3} = \eta_{3,I_4} = 0$ e $\eta_{3,I_2} = 1$.

Para $d \in \{4, \ldots, 9\}$, procedemos de modo análogo à situação anterior.

Se $d = 4$, então:

$$\eta_{4,k} = \sum_{d'=4+1}^{4+6} \sum_{k_{7-(d'-4)}=k} p_{d'} = \sum_{d'=5}^{10} \sum_{k_{11-d'}=k} p_{d'}.$$

Ora, $\eta_{4,k} = 0$ se $k_{11-d'} \neq k$ onde $d' = 7$, isto é, $k_{11-7} = k_4 = I_3$. Logo, $\eta_{4,I_3} = 1$ e $\eta_{4,I_1} = \eta_{4,I_2} = \eta_{4,I_4} = 0$.

Se $d = 5$, então:

$$\eta_{5,k} = \sum_{d'=5+1}^{5+6} \sum_{k_{7-(d'-5)}=k} p_{d'} = \sum_{d'=6}^{11} \sum_{k_{12-d'}=k} p_{d'}.$$

Ora, $\eta_{5,k} = 0$ se $k_{12-d'} \neq k$ onde $d' = 7$, isto é, $k_{12-7} = k_5 = I_3$. Logo, $\eta_{5,I_3} = 1$ e

$\eta_{5,I_1} = \eta_{5,I_2} = \eta_{5,I_4} = 0$.

Caso $d = 6$ temos:

$$\eta_{6,k} = \sum_{d'=6+1}^{6+6} \sum_{k_{7-(d'-6)}=k} p_{d'} = \sum_{d'=7}^{12} \sum_{k_{13-d'}=k} p_{d'}.$$

Então $\eta_{6,k} = 0$ se $k_{13-d'} \neq k$ onde $d' = 7$, isto é, $k_{13-7} = k_6 = I_4$. Logo, $\eta_{6,I_4} = 1$ e $\eta_{6,I_1} = \eta_{6,I_2} = \eta_{6,I_3} = 0$.

Para $d = 7, 8, 9$ temos, respetivamente:

$$\begin{aligned}
\eta_{7,k} &= \sum_{d'=7+1}^{7+6} \sum_{k_{7-(d'-7)}=k} p_{d'} = \sum_{d'=8}^{13} \sum_{k_{14-d'}=k} p_{d'} \\
\eta_{8,k} &= \sum_{d'=8+1}^{8+6} \sum_{k_{7-(d'-8)}=k} p_{d'} = \sum_{d'=9}^{14} \sum_{k_{15-d'}=k} p_{d'} \\
\eta_{9,k} &= \sum_{d'=9+1}^{9+6} \sum_{k_{7-(d'-9)}=k} p_{d'} = \sum_{d'=10}^{15} \sum_{k_{16-d'}=k} p_{d'}.
\end{aligned}$$

Nestes três casos, qualquer que seja a escolha de d' temos $p_{d'} = 0$. Então,

$$\eta_{d,\,k} = 0 \quad \forall\, d \in \{7, 8, 9\},\, \forall\, k \in \{I_1,\, I_2,\, I_3\}.$$

Assim:

$$\begin{aligned}
d &= 1 & d &= 2,\, d = 3 \\
y_{t_1,1,I_2} &= y_{t_1,1,I_3} = y_{t_1,1,I_4} = 0 & y_{t_1,d,I_1} &= y_{t_1,d,I_3} = y_{t_1,d,I_4} = 0 \\
y_{t_2,1,I_2} &= y_{t_2,1,I_3} = y_{t_2,1,I_4} = 0 & y_{t_2,d,I_1} &= y_{t_2,d,I_3} = y_{t_2,d,I_4} = 0 \\
y_{t_3,1,I_2} &= y_{t_3,1,I_3} = y_{t_3,1,I_4} = 0 & y_{t_3,d,I_1} &= y_{t_3,d,I_3} = y_{t_3,d,I_4} = 0 \\
y_{t_4,1,I_2} &= y_{t_4,1,I_3} = y_{t_4,1,I_4} = 0 & y_{t_4,d,I_1} &= y_{t_4,d,I_3} = y_{t_4,d,I_4} = 0 \\
d &= 4, 5 & d &= 6 \\
y_{t_1,d,I_1} &= y_{t_1,d,I_2} = y_{t_1,d,I_4} = 0 & y_{t_1,6,I_1} &= y_{t_1,6,I_2} = y_{t_1,6,I_3} = 0 \\
y_{t_2,d,I_1} &= y_{t_2,d,I_2} = y_{t_2,d,I_4} = 0 & y_{t_2,6,I_1} &= y_{t_2,6,I_2} = y_{t_2,6,I_3} = 0 \\
y_{t_3,d,I_1} &= y_{t_3,d,I_2} = y_{t_3,d,I_4} = 0 & y_{t_3,6,I_1} &= y_{t_3,6,I_2} = y_{t_3,6,I_3} = 0 \\
y_{t_4,d,I_1} &= y_{t_4,d,I_2} = y_{t_4,d,I_4} = 0 & y_{t_4,6,I_1} &= y_{t_4,6,I_2} = y_{t_4,6,I_3} = 0
\end{aligned}$$

e $\forall\, d \in \{7, 8, 9\}$,

$$y_{t_1,d,I_1} = y_{t_1,d,I_2} = y_{t_1,d,I_3} = y_{t_2,d,I_4} = 0$$

$$y_{t_2,d,I_1} = y_{t_2,d,I_2} = y_{t_2,d,I_3} = y_{t_2,d,I_4} = 0$$

$$y_{t_3,d,I_1} = y_{t_3,d,I_2} = y_{t_3,d,I_3} = y_{t_3,d,I_4} = 0$$

$$y_{t_4,d,I_1} = y_{t_4,d,I_2} = y_{t_4,d,I_3} = y_{t_4,d,I_4} = 0.$$

Apêndice B

Implementação em Python

Neste apêndice encontram-se os códigos implementados em *Python* dos casos das Secções 3.3, 4.2 e 4.3.

No *Python*, a indexação de uma lista com n elementos inicia-se em 0 e termina em $n-1$. Por esse motivo, foi necessário ajustar os valores dos índices cada vez que usássemos algum tipo de indexação.

Em cada código, estão presentes todos os dados relevantes – vetor das encomendas, número de prateleiras, configurações e tipos de prateleiras, e também da capacidade.

Algumas linhas do código contêm comentários que são usados para descrever o que o significado dos parâmetros e para perceber o que as funções retornam.

Para melhor referência, vamos explicitar na Tabela B.1 alguns dos comandos usados do pacote *Pyomo*:

Comando	**Breve descrição**
`ConcreteModel()`	Criação do modelo (vazio)
`Set(initialize=[xs])`	Cria o conjunto de índices a partir de `xs`
`Var(X, within = domain)`	Variável com índices no conjunto `X` e com domínio `domain`
`Objective(expr=X,sense=goal)`	Função objetivo com expressão `X`, com objetivo `goal = minimize` ou `maximize`
`ConstraintList()`	Criação da lista de restrições
`model.NAME.add(expr)`	Adição de `expr` à lista de restrições `model.NAME`
`SolverFactory('solver')`	Carregar o `solver` que vai ser usado
`solver.solve(modelo)`	Resolver o `modelo` com o `solver`

Tabela B.1: Descrição de algumas palavras-chave do pacote *Pyomo*.

B.1 Código – Caso Simples

```
#!/usr/bin/env python
# coding: utf-8

from pyomo.environ import*
import pyomo.opt as po

model = ConcreteModel(name="Caso_Simples")

Dbar = 10 # Dias
G = 6 # Dias de crescimento
Q = 5 # capacidade de cada prateleira

K = ['I1','I2','I3','I4'] # configuracao

#sucessao de configuracoes para o crescimento da planta
Kg = ['I1','I2','I2','I3','I3','I4']

Encomenda = [0,0,0,0,0,0,1,0,0,0]

# numero de prateleiras : 10
# T_i: conjunto de prateleiras de tipo i
T_1 = ['t1']
T_2 = ['t2']
T_3 = ['t3']
T_4 = ['t4']

# conjunto de todas as prateleiras
# em que as plantas crescem
T = ['t1','t2','t3','t4']

# conjunto das prateleiras (T) com o banco de sementes (S, sigma)
# e com o armazem (A,tau)
T_1 = ['S','t1','t2','t3','t4','A']

# pares [prateleira,configuracao admissivel]
TK = [ ['t1','I1'],['t2','I2'], \
       ['t3','I3'],['t4','I4'] ]
```

```
38
39 def pair_shelf(xs1,xs2):
40     parset=[]
41     for i in range(len(xs1)):
42         for j in range(i,len(xs2)):
43             parset.append([xs1[i],xs2[j]])
44     return parset
45
46 # lista de todos os 4-tuplos correspondentes
47 # aos 4 indices das variaveis x
48 def xvar(g=G,xs=pair_shelf(T_1,T_1),dbar=Dbar):
49     tripset=[]
50     for pdia in range(dbar):
51         for gdia in range(g+1):
52             for k in xs:
53                 tripset.append([gdia,k,pdia])
54     return tripset
55
56 # lista de todos os 3-tuplos correspondentes
57 # aos 3 indices das variaveis y
58 def yvar(xs=T,dbar=Dbar,k=K):
59     ylist=[]
60     for u in range(len(xs)):
61         for pdia in range(1,dbar):
62             for v in range(len(k)):
63                 ylist.append([xs[u],pdia,k[v]])
64     return ylist
65
66 # com as listas construidas para os indices das
67 # variaveis x e y, definimos todas as variaveis
68 model.xset=Set(initialize=xvar())
69 model.x=Var(model.xset,within=NonNegativeIntegers)
70
71 model.yset=Set(initialize=yvar())
72 model.y=Var(model.yset,within=Binary)
73
74 ### Funcao Objetivo
75 # na funcao objetivo entram todas as variaveis x[g,si,sj,d]
76 # em que si e sj sao diferentes, e tanto si como sj nao sao
```

```
77  # nem S (sigma,banco de sementes) nem A (tau,armazem)
78
79  xs1 = []
80  xs2 = []
81  xs3 = []
82
83  for dia in range(Dbar):
84      for g in range(G+1):
85          for i in range(len(T_l)):
86              xs1.append(model.x[g,T_l[i],T_l[i],dia])
87          for i in range(len(T)):
88              xs2.append(model.x[g,'S',T[i],dia])
89              xs2.append(model.x[g,T[i],'A',dia])
90          xs3.append(model.x[g,'S','A',dia])
91
92  model.Objetivo = Objective(expr=sum(model.x[i] for i in model.x) \
93                          -sum(xs1)-sum(xs2)-sum(xs3), sense=minimize)
94
95  ### Restricoes
96  ##Procura
97  model.Procura = ConstraintList()
98
99  def procura(shelf, dia):
100     soma = []
101     for i in range(len(shelf)):
102         soma.append(model.x[G, shelf[i], 'A', dia - 1 ])
103     return sum(soma)
104
105 # introducao das restricoes
106 for dia in range(1,Dbar+1):
107     model.Procura.add(procura(T,dia) == Encomenda[dia - 1])
108
109 ## Capacidade
110 model.Capacidade = ConstraintList()
111
112 def capac_fixa(shelf,conf,Q):
113     x = [shelf,conf]
114     if x in TK:
115         return Q
```

```
116     else:
117         return 0
118 
119 def capacidade(shelf_i,conf,dia):
120     ind = T_l.index(shelf_i)
121     xs = []
122     ys = []
123     for g in range(1,G+1):
124         for shelf_f in range(ind,len(T_l)):
125             if Kg[g-1] == conf:
126                 xs.append(model.x[g,shelf_i,T_l[shelf_f], dia ])
127             else:
128                 ys.append(0)
129     return sum(xs)
130 
131 # introducao das restricoes
132 for dia in range(1, Dbar):
133     for si in range(len(T)):
134         for k in range(len(K)):
135             model.Capacidade.add(capacidade(T[si], K[k], dia) \
136                     <= capac_fixa(T[si],K[k],Q)*model.y[T[si], dia, K[k]])
137 
138 ## Plantacao
139 model.Plantacao = ConstraintList()
140 
141 def plantacao(shelf, dia):
142     soma = []
143     for i in range(len(shelf)):
144         soma.append(model.x[0, 'S', shelf[i], dia - G - 1 ])
145     return sum(soma)
146 
147 # introducao das restricoes
148 for d in range(G+1,Dbar+1):
149     model.Plantacao.add(plantacao(T,d) == Encomenda[d - 1])
150 
151 ## Configuracao
152 model.Configuracao = ConstraintList()
153 
154 def configuracao(shelf,dia,k):
```

```
155     soma = []
156     for i in range(len(k)):
157         soma.append(model.y[shelf,dia,k[i]])
158     return sum(soma)
159 
160 # introducao das restricoes
161 for dia in range(1,Dbar):
162     for shelf in range(len(T)):
163         model.Configuracao.add(configuracao(T[shelf],dia,K) <= 1)
164 
165 ## Equilibrio
166 model.Equilibrio = ConstraintList()
167 
168 # Membro Direito
169 def equil_D(shelf,shelf_f, dia, g):
170     somaR = []
171     iT = shelf_f.index(shelf)
172     for j in range(iT,len(shelf_f)):
173         somaR.append(model.x[g+1,shelf,shelf_f[j],dia])
174     return sum(somaR)
175 
176 #Membro Esquerdo
177 def equil_E(shelf,shelf_i, dia, g):
178     somaL = []
179     iT = shelf_i.index(shelf)
180     for j in range(len(shelf_i)):
181         if j < iT+1:
182             somaL.append(model.x[g,shelf_i[j],shelf,dia -1])
183     return sum(somaL)
184 
185 # introducao das restricoes
186 for dia in range(1,Dbar):
187     for g in range(G):
188         for shelf in range(len(T)):
189             model.Equilibrio.add( equil_E(T[shelf],T_l, dia, g) \
190                                 == equil_D(T[shelf], T_l, dia, g) )
191 
192 ### Fixacao de Variaveis
193 ## FV1
```

```
194 model.FV_1 = ConstraintList()
195
196 for dia in range(Dbar):
197     for g in range(G+1):
198         model.FV_1.add(model.x[g,'S','A', dia] == 0)
199         model.FV_1.add(model.x[g,'S','S', dia] == 0)
200         model.FV_1.add(model.x[g,'A','A', dia] == 0)
201
202 ## FV2
203 model.FV_2 = ConstraintList()
204
205 for dia in range(Dbar):
206     for g in range(1,G+1):
207         for shelf in range(len(T_l)):
208             model.FV_2.add(model.x[g, 'S', T_l[shelf], dia] == 0)
209
210     for g in range(G):
211         for shelf in range(len(T_l)):
212             model.FV_2.add(model.x[g, T_l[shelf], 'A', dia] == 0)
213
214 dia_0_G = [0, G]
215
216 for dia in range(1,Dbar):
217     for g in range(len(dia_0_G)):
218         for si in range(len(T)):
219             for sf in range(si,len(T)):
220                 model.FV_2.add(model.x[dia_0_G[g], T[si], T[sf],dia] == 0 )
221
222 ## FV3
223 model.FV_3 = ConstraintList()
224
225 for dia in range(Dbar):
226     for shelf in range(1,len(T_l)):
227         for s in range(shelf,len(T_l)):
228             model.FV_3.add(model.x[0, T_l[shelf], T_l[s], dia] == 0)
229
230 for dia in range(Dbar):
231     for shelf in range(len(T_l)-1):
232         for s in range(shelf,len(T_l)-1):
```

```
233             model.FV_3.add(model.x[G, T_l[shelf], T_l[s], dia] == 0)
234
235 ## FV4
236 model.FV_4 = ConstraintList()
237
238 def ultimo_dia_procura(lista):
239     xs = []
240     for i in range(len(lista)):
241         if lista[i]!=0:
242             xs.append(i)
243         else:
244             continue
245     return max(xs)+1
246
247 dia_l = ultimo_dia_procura(Encomenda)
248 ult_dia_poss = dia_l - G + 1
249
250 for dia in range(ult_dia_poss, dia_l + 1):
251     for g in range(G - (dia_l - dia) +1):
252         for shelf in range(len(T_l)):
253             for s in range(shelf,len(T_l)):
254                 model.FV_4.add(model.x[g, T_l[shelf], T_l[s], dia] == 0)
255
256 for shelf in range(len(T)):
257     model.FV_4.add(model.x[0,'S',T[shelf],Dbar-1] ==0)
258
259 ## FV5
260 model.FV_5 = ConstraintList()
261
262 for dia in range(Dbar):
263     if dia < G:
264         for g in range(dia+1, G+1):
265             for si in range(len(T_l)):
266                 for sf in range(si, len(T_l)):
267                     model.FV_5.add(model.x[g, T_l[si], T_l[sf], dia] == 0)
268     else: break
269
270 ## FV6
271 model.FV_6 = ConstraintList()
```

```
272
273 for dia in range(G+1, Dbar+1):
274     for shelf in range(len(T)):
275         model.FV_6.add(model.x[0, 'S', T[shelf], dia-G-1] \
276                        <= Encomenda[dia-1])
277
278 for dia in range(1,Dbar+1):
279     for shelf in range(len(T)):
280         model.FV_6.add(model.x[G, T[shelf], 'A', dia-1] \
281                        <= Encomenda[dia-1])
282
283 ## FV7
284 model.FV_7 = ConstraintList()
285
286 def soma_procura(dia, k):
287     x1 = []
288     for dl in range(dia +1, dia + G+1):
289         if dl <= len(Encomenda):
290             dk = G + dia +1 - dl
291             if Kg[dk-1] == k:
292                 x1.append(Encomenda[dl-1])
293             else:
294                 continue
295         else:
296             break
297     if x1==[]:
298         x1.append(0)
299     return x1
300
301 def eta_dk():
302     soma=[]
303     for dia in range(1,Dbar):
304         soma_I1 = soma_procura(dia,'I1')
305         soma_I2 = soma_procura(dia,'I2')
306         soma_I3 = soma_procura(dia,'I3')
307         soma_I4 = soma_procura(dia,'I4')
308         soma.append([sum(soma_I1),sum(soma_I2),\
309                      sum(soma_I3),sum(soma_I4)])
310     return soma
```

```
311
312 for dia in range(1,Dbar):
313     for k in range(len(K)):
314         if eta_dk()[dia-1][k]==0:
315             for shelf in range(len(T)):
316                 model.FV_7.add(model.y[T[shelf],dia,K[k]]==0)
317         else:
318             continue
319
320 ### Solucao
321 opt=SolverFactory('glpk')
322 solucao = opt.solve(model,tee=True)
323 model.Objetivo.display() # mostrar o valor do objetivo
324
325 #aceder aos valores das variaveis x e y
326 for i in model.x:
327     if model.x[i].value != 0 :
328         print(str(model.x[i]), model.x[i].value)
329 for i in model.y:
330     if model.y[i].value != 0:
331         print(str(model.y[i]), model.y[i].value)
332
```

B.2 Código – Caso com 100 dias

```
#!/usr/bin/env python
# coding: utf-8

from pyomo.environ import*
import pyomo.opt as po

model = ConcreteModel(name="Caso_Simples")

Dbar = 100 # Dias
G = 20 #Dias de crescimento
Q = 12 # capacidade de cada prateleira

K = ['I1','I2','I3'] # configuracao

Encomenda = [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  \
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  \
            0, 0, 0, 5, 8, 12, 0, 0, 0, 0, \
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  \
            0, 0, 0, 0, 0, 0, 0, 0, 18, 0, \
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  \
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  \
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  \
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  \
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0   ]

# numero de prateleiras : 10
# T_i: conjunto de prateleiras de tipo i
T_1 = ['s1','s2','s3']
T_2 = ['s4','s5','s6','s7','s8']
T_3 = ['s9','s10']

# T: conjunto dos tipos de prateleiras
T = [T_1,T_2,T_3]

# S: conjunto de todas as prateleiras
# em que as plantas crescem
S = ['s1','s2','s3','s4','s5', \
```

```
      's6','s7','s8','s9','s10']

# S_l: conjunto das prateleiras (S) com o banco de sementes ('S')
# e com o armazem ('A')
S_l = ['S','s1','s2','s3','s4','s5',\
      's6','s7','s8','s9','s10','A']

#sucessao de configuracoes para o crescimento da planta
Kg = ['I1','I1','I1','I1','I1','I1','I2','I2','I2','I2', \
      'I2','I2','I2','I2','I2','I2','I2','I2','I3','I3']

# pares [prateleira,configuracao admissivel]
TK = [ ['s1','I1'],['s2','I1'],['s3','I1'], \
       ['s4','I2'],['s5','I2'],['s6','I2'],['s7','I2'],['s8','I2'], \
       ['s9','I3'],['s10','I3'] ]

# lista dos movimentos possiveis
def total_pairs():
    xs = []
    for i1 in range(len(S_l)):
        xs.append([S_l[i1],S_l[i1]])
    for i2 in range(1,len(S_l)):
        xs.append(['S',S_l[i2]])
    for i3 in range(1,len(S_l)-1):
        xs.append([S_l[i3],'A'])

    for j1 in range(len(T_1)):
        for j12 in range(len(T_2)):
            xs.append([T_1[j1],T_2[j12]])
        for j13 in range(len(T_3)):
            xs.append([T_1[j1],T_3[j13]])

    for j2 in range(len(T_2)):
        for j23 in range(len(T_3)):
            xs.append([T_2[j2],T_3[j23]])
    return xs

# lista de todos os 4-tuplos correspondentes
# aos 4 indices das variaveis x
```

```
77  def xvar(g=G,xs=total_pairs(),dbar=Dbar):
78      tripset=[]
79      for pdia in range(dbar):
80          for gdia in range(g+1):
81              for k in xs:
82                  tripset.append([gdia,k,pdia])
83      return tripset
84
85  # lista de todos os 3-tuplos correspondentes
86  # aos 3 indices das variaveis y
87  def yvar(xs=S,dbar=Dbar,k=K):
88      ylist=[]
89      for u in range(len(xs)):
90          for pdia in range(1,dbar):
91              for v in range(len(k)):
92                  ylist.append([xs[u],pdia,k[v]])
93      return ylist
94
95  # com as listas construidas para os indices das
96  # variaveis x e y, definimos todas as variaveis
97  model.xset=Set(initialize=xvar())
98  model.x=Var(model.xset,within=NonNegativeIntegers)
99
100 model.yset=Set(initialize=yvar())
101 model.y=Var(model.yset,within=Binary)
102
103 ### Funcao Objetivo
104 # na funcao objetivo entram todas as variaveis x[g,si,sj,d]
105 # em que si e sj sao diferentes, e tanto si como sj nao sao
106 # nem S (sigma,banco de sementes) nem A (tau,armazem)
107
108 xs1 = []
109 xs2 = []
110 xs3 = []
111
112 for dia in range(Dbar):
113     for g in range(G+1):
114         for i in range(len(S_l)):
115             xs1.append(model.x[g,S_l[i],S_l[i],dia])
```

```
116         for i in range(len(S)):
117             xs2.append(model.x[g,'S',S[i],dia])
118             xs2.append(model.x[g,S[i],'A',dia])
119         xs3.append(model.x[g,'S','A',dia])
120 
121 #Objetivo
122 model.Objetivo = Objective(expr=sum(model.x[i] for i in model.x) \
123                         -sum(xs1)-sum(xs2)-sum(xs3), sense=minimize)
124 
125 ### Restricoes
126 ## Procura
127 model.Procura = ConstraintList()
128 
129 def procura(shelf, dia):
130     soma = []
131     for i in range(len(shelf)):
132         soma.append(model.x[G, shelf[i], 'A', dia - 1 ])
133     return sum(soma)
134 
135 # introducao das restricoes
136 for dia in range(1,Dbar+1):
137     model.Procura.add(procura(S,dia) == Encomenda[dia - 1])
138 
139 ## Capacidade
140 model.Capacidade = ConstraintList()
141 
142 def capac_fixa(shelf,conf,Q):
143     x = [shelf,conf]
144     if x in TK:
145         return Q
146     else:
147         return 0
148 
149 def find_ind_set(xs):
150     ''' Encontrar os indices das prateleiras de cada tipo T em S_l
151     >>> find_ind_set(T_2)
152     [4, 5, 6, 7, 8]
153     '''
154     x = []
```

```
155     for i in range(len(xs)):
156         x.append(S_l.index(xs[i]))
157     return x
158 
159 def ind_listL(shelf,xs):
160     ''' Indices das prateleiras que saem de shelf
161     >>> ind_listL('s5',T_2)
162     [5, 9, 10, 11]
163     '''
164     x1 = []
165     x2 = find_ind_set(xs)
166     x1.append(S_l.index(shelf))
167     for i in range(max(x2)+1,len(S_l)):
168         x1.append(i)
169     x1.sort()
170     return x1
171 
172 def capacidade(shelf_i,xs,conf,dia):
173     ind = ind_listL(shelf_i,xs)
174     xs = []
175     ys = []
176     for g in range(1,G+1):
177         for shelf_f in range(len(ind)):
178             if Kg[g-1] == conf:
179                 xs.append(model.x[g,shelf_i,S_l[ind[shelf_f]], dia ])
180             else:
181                 ys.append(0)
182     return sum(xs)
183 
184 # introducao das restricoes
185 for dia in range(1, Dbar):
186     for k in range(len(K)):
187         for si in range(len(T_1)):
188             model.Capacidade.add(capacidade(T_1[si],T_1, K[k], dia) \
189                     <= capac_fixa(T_1[si],K[k],Q)*model.y[T_1[si], dia, K[k]])
190         for si in range(len(T_2)):
191             model.Capacidade.add(capacidade(T_2[si],T_2, K[k], dia) \
192                     <= capac_fixa(T_2[si],K[k],Q)*model.y[T_2[si], dia, K[k]])
193         for si in range(len(T_3)):
```

```
194             model.Capacidade.add(capacidade(T_3[si],T_3, K[k], dia) \
195                     <= capac_fixa(T_3[si],K[k],Q)*model.y[T_3[si], dia, K[k]])
196
197 ## Plantacao
198 model.Plantacao = ConstraintList()
199
200 def plantacao(shelf, dia):
201     soma = []
202     for i in range(len(shelf)):
203         soma.append(model.x[0, 'S', shelf[i], dia - G - 1 ])
204     return sum(soma)
205
206 # introducao das restricoes
207 for d in range(G+1,Dbar+1):
208     model.Plantacao.add(plantacao(S,d) == Encomenda[d - 1])
209
210
211 ## Configuracao
212 model.Configuracao = ConstraintList()
213
214 def configuracao(shelf_set,dia,k):
215     soma = []
216     for i in range(len(shelf_set)):
217         for j in range(len(k)):
218             soma.append(model.y[shelf_set[i],dia,k[j]])
219     return sum(soma)
220
221 # introducao das restricoes
222 for dia in range(1,Dbar):
223     for shelf_type in range(len(T)):
224         model.Configuracao.add(configuracao(T[shelf_type],dia,K) \
225                         <= len(T[shelf_type]))
226
227 ## Equilibrio
228 model.Equilibrio = ConstraintList()
229
230 def find_ind_set(xs):
231     ''' Encontrar os indices das prateleiras de cada tipo T em S_l'''
232     x = []
```

```
233     for i in range(len(xs)):
234         x.append(S_l.index(xs[i]))
235     return x
236 
237 # Membro esquerdo
238 def ind_listE(shelf,xs):
239     ''' Indices das prateleiras com destino shelf
240     >>> ind_listE('s4',T_2)
241     [0, 1, 2, 3, 4] # 'S','s1','s2','s3','s4'
242     '''
243     x1 = []
244     x2 = find_ind_set(xs)
245     x1.append(S_l.index(shelf))
246     for i in range(min(x2)):
247         x1.append(i)
248     x1.sort()
249     return x1
250 
251 def rec_el_E(shelf,xs,g,dia):
252     ''' pares corretos com destino shelf '''
253     x1 = ind_listE(shelf,xs)
254     xs = []
255     for i in range(len(x1)):
256         xs.append(model.x[g,S_l[x1[i]],shelf,dia -1])
257     return sum(xs)
258 
259 # Membro direito
260 def ind_listD(shelf,xs):
261     ''' Indices das prateleiras que saem de shelf
262     >>> ind_listD('s4',T_2)
263     [4, 9, 10, 11] # 's4','s9','s10','A'
264     '''
265     x1 = []
266     x2 = find_ind_set(xs)
267     x1.append(S_l.index(shelf))
268     for i in range(max(x2)+1,len(S_l)):
269         x1.append(i)
270     x1.sort()
271     return x1
```

```
272
273 def rec_el_D(shelf,xs,g,dia):
274     ''' pares corretos provenientes de shelf '''
275     x1 = ind_listD(shelf,xs)
276     xs = []
277     for i in range(len(x1)):
278         xs.append(model.x[g+1,shelf,S_l[x1[i]],dia])
279     return sum(xs)
280
281 # introducao das restricoes
282 for dia in range(1,Dbar):
283     for g in range(G): # G+1 ou G
284         for s1 in range(len(T_1)):
285             model.Equilibrio.add( rec_el_E(T_1[s1],T_1, g, dia) \
286                                 == rec_el_D(T_1[s1], T_1, g, dia) )
287
288         for s2 in range(len(T_2)):
289             model.Equilibrio.add( rec_el_E(T_2[s2],T_2, g, dia) \
290                                 == rec_el_D(T_2[s2], T_2, g, dia) )
291
292         for s3 in range(len(T_3)):
293             model.Equilibrio.add( rec_el_E(T_3[s3],T_3, g, dia) \
294                                 == rec_el_D(T_3[s3], T_3, g, dia) )
295
296 ### Fixacao de Variaveis
297 ## FV1
298 model.FV_1 = ConstraintList()
299
300 for dia in range(Dbar):
301     for g in range(G+1):
302         model.FV_1.add(model.x[g,'S','A',dia] == 0)
303         model.FV_1.add(model.x[g,'S','S',dia] == 0)
304         model.FV_1.add(model.x[g,'A','A',dia] == 0)
305
306 ## FV2
307 model.FV_2 = ConstraintList()
308
309 for dia in range(Dbar):
310     for g in range(1,G+1):
```

```
311         for shelf in range(len(S_1)):
312             model.FV_2.add(model.x[g, 'S', S_1[shelf],dia] == 0)
313 
314     for g in range(G):
315         for shelf in range(len(S_1)):
316             model.FV_2.add(model.x[g, S_1[shelf], 'A',dia] == 0)
317 
318 dia_0_G = [0, G]
319 
320 for dia in range(1,Dbar):
321     for g in range(len(dia_0_G)):
322 
323         for s1 in range(len(T_1)):
324             for s2 in range(len(T_2)):
325                 model.FV_2.add(model.x[dia_0_G[g], T_1[s1], T_2[s2],dia] == 0)
326 
327             for s3 in range(len(T_3)):
328                 model.FV_2.add(model.x[dia_0_G[g], T_1[s1], T_3[s3],dia] == 0)
329 
330         for s2 in range(len(T_2)):
331             for s3 in range(len(T_3)):
332                 model.FV_2.add(model.x[dia_0_G[g], T_2[s2], T_3[s3],dia] == 0)
333 
334         for shelf in range(len(S)):
335             model.FV_2.add(model.x[dia_0_G[g], S[shelf], S[shelf],dia] == 0)
336 
337 ## FV3
338 model.FV_3 = ConstraintList()
339 
340 for dia in range(Dbar):
341     for sh1 in range(len(T_1)):
342         for sh2 in range(1+len(T_1)+1,len(S_1)):
343             model.FV_3.add(model.x[0, T_1[sh1], S_1[sh2],dia] == 0)
344 
345     for sh2 in range(len(T_2)):
346         for sh3 in range(1+len(T_1)+len(T_2)+1,len(S_1)):
347             model.FV_3.add(model.x[0, T_2[sh2], S_1[sh3],dia] == 0)
348 
349     for shelf in range(len(S)):
```

```
        model.FV_3.add(model.x[0, S[shelf], S[shelf],dia] == 0)

for dia in range(Dbar):
    for sh in range(1,len(S_l)-1):
        model.FV_3.add(model.x[G, 'S', S_l[sh],dia] == 0)

    for sh1 in range(len(T_1)):
        for sh2 in range(1+len(T_1)+1,len(S_l)-1):
            model.FV_3.add(model.x[G, T_1[sh1], S_l[sh2],dia] == 0)

    for sh2 in range(len(T_2)):
        for sh3 in range(1+len(T_1)+len(T_2)+1,len(S_l)-1):
            model.FV_3.add(model.x[G, T_2[sh2], S_l[sh3],dia] == 0)

    for shelf in range(len(S)):
        model.FV_3.add(model.x[G, S[shelf], S[shelf],dia] == 0)

## FV4
model.FV_4 = ConstraintList()

def ultimo_dia_procura(lista):
    xs = []
    for i in range(len(lista)):
        if lista[i]!=0:
            xs.append(i)
        else:
            continue
    return max(xs)+1

dia_l = ultimo_dia_procura(Encomenda)
ult_dia_P = dia_l - G + 1

for dia in range(ult_dia_P, dia_l + 1):
    for g in range(G - (dia_l - dia) +1):

        for sh1 in range(len(T_1)):
            for sh2 in range(len(T_2)):
                model.FV_4.add(model.x[g, T_1[sh1], T_2[sh2],dia] == 0)

```

```
389             for sh3 in range(len(T_3)):
390                 model.FV_4.add(model.x[g, T_1[sh1], T_3[sh3],dia] == 0)
391 
392         for sh2 in range(len(T_2)):
393             for sh3 in range(len(T_3)):
394                 model.FV_4.add(model.x[g, T_2[sh2], T_3[sh3],dia] == 0)
395 
396         for shelf in range(len(S)):
397             model.FV_4.add(model.x[g, S[shelf], S[shelf],dia] == 0)
398 
399 for shelf in range(len(S)):
400     model.FV_4.add(model.x[0,'S',S[shelf],Dbar-1] == 0)
401 
402 ## FV5
403 model.FV_5 = ConstraintList()
404 
405 for dia in range(Dbar):
406     if dia < G:
407         for g in range(dia+1, G+1):
408             for sh in range(1,len(S_1)):
409                 model.FV_5.add(model.x[G, 'S', S_1[sh],dia] == 0)
410 
411             for sh1 in range(len(T_1)):
412                 for sh2 in range(1+len(T_1)+1,len(S_1)):
413                     model.FV_5.add(model.x[G, T_1[sh1], S_1[sh2],dia] == 0)
414 
415             for sh2 in range(len(T_2)):
416                 for sh3 in range(1+len(T_1)+len(T_2)+1,len(S_1)):
417                     model.FV_5.add(model.x[G, T_2[sh2], S_1[sh3],dia] == 0)
418 
419             for shelf in range(len(S)):
420                 model.FV_5.add(model.x[G, S[shelf], S[shelf],dia] == 0)
421 
422 ## FV6
423 model.FV_6 = ConstraintList()
424 
425 for dia in range(G+1, Dbar+1):
426     for shelf in range(len(S)):
427         model.FV_6.add(model.x[0, 'S', S[shelf], dia-G-1] \
```

```
                    <= Encomenda[dia-1])

for dia in range(1,Dbar+1):
    for shelf in range(len(S)):
        model.FV_6.add(model.x[G, S[shelf], 'A', dia-1] \
                    <= Encomenda[dia-1])

## FV7
model.FV_7 = ConstraintList()

def soma_procura(dia, k):
    x1 = []
    for dl in range(dia +1, dia + G+1):
        if dl <= len(Encomenda):
            dk = G + dia +1 - dl
            if Kg[dk-1] == k:
                x1.append(Encomenda[dl-1])
            else:
                continue
        else:
            break
    if x1==[]:
        x1.append(0)
    return x1

def eta_dk():
    soma=[]
    for dia in range(1,Dbar):
        soma_I1 = soma_procura(dia,'I1')
        soma_I2 = soma_procura(dia,'I2')
        soma_I3 = soma_procura(dia,'I3')
        soma.append([sum(soma_I1),sum(soma_I2),sum(soma_I3)])
    return soma

for dia in range(1,Dbar):
    for k in range(len(K)):
        if eta_dk()[dia-1][k]==0:
            for shelf in range(len(S)):
                model.FV_7.add(model.y[S[shelf],dia,K[k]]==0)
```

```
467         else: continue
468
469 ### Solucao
470 opt=SolverFactory('glpk')
471 solucao = opt.solve(model,tee=True)
472 model.Objetivo.display() # mostrar o valor do objetivo
473
474 # aceder aos valores das variaveis
475 for i in model.x:
476     if model.x[i].value != 0 :
477         print(str(model.x[i]), model.x[i].value)
478
479 for i in model.y:
480     if model.y[i].value != 0:
481         print(str(model.y[i]), model.y[i].value)
482
```

B.3 Código – Caso com 2 plantas

```
#!/usr/bin/env python
# coding: utf-8

from pyomo.environ import*
import pyomo.opt as po

model = ConcreteModel(name="Caso_Simples")

#Horizonte de planeamento
Dbar = 15 # Dias

C = ['c1','c2'] #crops

#tempo de crescimento de cada planta
G1, G2 = 8, 10 #c1, c2
Gc = [8,10] # [G1,G2]

Q = 5 # capacidade de cada prateleira

# configuracao
K = ['I1','I2','I3','I4','I5']

#sucessao de configuracoes para o crescimento de cada planta
# Kg = [Kgc1, Kgc2 ]
Kg = [['I1','I2','I2','I3','I3','I3','I3','I4'], \
      ['I1','I1','I2','I2','I2','I3','I3','I3','I4','I5'] ]

# Encomenda = [[Encomenda c1],[Encomenda c2]]
Encomenda = [[0,0,0,0,0,0,0,0,0,0,0,2,0,0,0], \
             [0,0,0,0,0,0,0,0,0,0,0,0,3,0,0] ]

# numero de prateleiras : 16
# T_i: conjunto de prateleiras de tipo i
T_1 = ['s1','s2','s3']          # prateleiras do tipo T_1
T_2 = ['s4','s5','s6','s7']     # prateleiras do tipo T_2
T_3 = ['s8','s9','s10','s11']   # prateleiras do tipo T_3
T_4 = ['s12','s13','s14']       # prateleiras do tipo T_4
```

```
38 T_5 = ['s15','s16']              # prateleiras do tipo T_5
39
40 # T: conjunto dos tipos de prateleiras
41 T = [T_1,T_2,T_3,T_4,T_5]
42
43 # S: conjunto de todas as prateleiras
44 # em que as plantas crescem
45 S = ['s1','s2','s3','s4','s5','s6','s7','s8', \
46      's9','s10','s11','s12','s13','s14','s15','s16']
47
48 # S_l: conjunto das prateleiras (S) com o
49 # banco de sementes ('S',sigma)
50 # e com o armazem ('A',tau)
51 S_l = ['S','s1','s2','s3','s4','s5','s6','s7','s8',\
52       's9','s10','s11','s12','s13','s14','s15','s16','A']
53
54 # pares [prateleira,configuracao admissivel]
55 TK = [ ['s1','I1'],['s2','I1'],['s3','I1'], \
56        ['s4','I2'],['s5','I2'],['s6','I2'],['s7','I2'], \
57        ['s8','I3'],['s9','I3'],['s10','I3'],['s11','I3'], \
58        ['s12','I4'], ['s13','I4'], ['s14','I4'], \
59        ['s15','I5'], ['s16','I5'] ]
60
61 # lista dos movimentos possiveis
62 def total_pairs():
63     xs = []
64     for i1 in range(len(S_l)):
65         xs.append([S_l[i1],S_l[i1]])
66
67     for i2 in range(1,len(S_l)):
68         xs.append(['S',S_l[i2]])
69
70     for i3 in range(1,len(S_l)-1):
71         xs.append([S_l[i3],'A'])
72
73     for j1 in range(len(T_1)):
74         for j12 in range(len(T_2)):
75             xs.append([T_1[j1],T_2[j12]])
76         for j13 in range(len(T_3)):
```

```
77              xs.append([T_1[j1],T_3[j13]])
78          for j14 in range(len(T_4)):
79              xs.append([T_1[j1],T_4[j14]])
80          for j15 in range(len(T_5)):
81              xs.append([T_1[j1],T_5[j15]])
82
83      for j2 in range(len(T_2)):
84          for j23 in range(len(T_3)):
85              xs.append([T_2[j2],T_3[j23]])
86          for j24 in range(len(T_4)):
87              xs.append([T_2[j2],T_4[j24]])
88          for j25 in range(len(T_5)):
89              xs.append([T_2[j2],T_5[j25]])
90
91      for j3 in range(len(T_3)):
92          for j34 in range(len(T_4)):
93              xs.append([T_3[j3],T_4[j34]])
94          for j35 in range(len(T_5)):
95              xs.append([T_3[j3],T_5[j35]])
96
97      for j4 in range(len(T_4)):
98          for j45 in range(len(T_5)):
99              xs.append([T_4[j4],T_5[j45]])
100     return xs
101
102 # lista de todos os 5-tuplos correspondentes
103 # aos 5 indices das variaveis x
104 def xvar():
105     xs=[]
106     xlista = total_pairs()
107     for crop in range(len(C)):
108         for pdia in range(Dbar):
109             for gdia in range(Gc[crop]+1):
110                 for k in range(len(xlista)):
111                     xs.append([C[crop],gdia,xlista[k],pdia])
112     return xs
113
114 # lista de todos os 3-tuplos correspondentes
115 # aos 3 indices das variaveis y
```

```
116 def yvar(xs=S,dbar=Dbar,k=K):
117     ylist=[]
118     for u in range(len(xs)):
119         for pdia in range(1,dbar):
120             for v in range(len(k)):
121                 ylist.append([xs[u],pdia,k[v]])
122     return ylist
123 
124 # com as listas construidas para os indices das
125 # variaveis x e y, definimos todas as variaveis
126 model.xset=Set(initialize=xvar())
127 model.x=Var(model.xset,within=NonNegativeIntegers)
128 
129 model.yset=Set(initialize=yvar())
130 model.y=Var(model.yset,within=Binary)
131 
132 ### Funcao Objetivo
133 # na funcao objetivo entram todas as variaveis x[c,g,si,sj,d]
134 # em que si e sj sao diferentes, e tanto si como sj nao sao
135 # nem S (banco de sementes) nem A (armazem)
136 xs1 = []
137 xs2 = []
138 xs3 = []
139 
140 for crop in range(len(C)):
141     for dia in range(Dbar):
142         for g in range(Gc[crop]+1):
143             for i in range(len(S_l)):
144                 xs1.append(model.x[C[crop],g,S_l[i],S_l[i],dia])
145 
146             for i in range(len(S)):
147                 xs2.append(model.x[C[crop],g,'S',S[i],dia])
148                 xs2.append(model.x[C[crop],g,S[i],'A',dia])
149 
150             xs3.append(model.x[C[crop],g,'S','A',dia])
151 
152 #Objetivo
153 model.Objetivo = Objective(expr=sum(model.x[i] for i in model.x) \
154                         -sum(xs1)-sum(xs2)-sum(xs3), sense=minimize)
```

```
155
156 ### Restricoes
157 ## Procura
158 model.Procura = ConstraintList()
159
160 def procura(shelf, dia):
161     soma = []
162     for i in range(len(shelf)):
163         soma.append(model.x[C[crop],Gc[crop], shelf[i], 'A', dia - 1 ])
164     return sum(soma)
165
166 # introducao das restricoes
167 for crop in range(len(C)):
168     for dia in range(1,Dbar+1):
169         model.Procura.add(procura(S,dia) == Encomenda[crop][dia - 1])
170
171 ## Capacidade
172 model.Capacidade = ConstraintList()
173
174 def capac_fixa(shelf,conf,Q):
175     ''' retorna Q se a prateleira (shelf)
176     pode estar na configuracao (conf) '''
177     x = [shelf,conf]
178     if x in TK:
179         return Q
180     else:
181         return 0
182
183 def find_ind_set(xs):
184     ''' Encontrar os indices das prateleiras de cada tipo T em S_l
185     >>> find_ind_set(T_1)
186     [1,2,3]
187     >>> ind_ind_set(T_4)
188     [12,13,14]
189     '''
190     x = []
191     for i in range(len(xs)):
192         x.append(S_l.index(xs[i]))
193     return x
```

```
194
195 def ind_listD(shelf,xs):
196     ''' Indices das prateleiras que saem de shelf
197     >>> ind_listD('s8',T_3)
198     [8, 12, 13, 14, 15, 16, 17]
199     '''
200     x1 = []
201     xn = []
202     x2 = find_ind_set(xs)
203     M = max(x2)
204     if not(shelf in xs):
205         return xn
206     else:
207         x1.append(S_l.index(shelf))
208         for i in range(M+1,len(S_l)):
209             x1.append(i)
210         x1.sort()
211     return x1
212
213 def find_ind_K(crop,conf):
214     ''' Encontra os dias de crescimento g tais que
215     Kg[crop][g]=conf
216     C[0] = 'c1'
217     >>> find_ind_K(0,'I3')
218     [4, 5, 6, 7]
219     '''
220     i1 = C.index(C[crop])
221     x = []
222     xa = []
223     xn = []
224     for i in range(1,Gc[crop]+1):
225         if Kg[i1][i-1] == conf:
226             x.append(i)
227         else:
228             xa.append(0)
229     if not x:
230         return xn
231     return x
232
```

```
233 def capacidade_c1(shelf_i,xs1,conf,dia):
234     xs = []
235     ind = ind_listD(shelf_i,xs1)
236     x1 = find_ind_K(0,conf)
237     if len(ind) != 0:
238         for shelf_f in range(len(ind)):
239             if not x1:
240                 return 0
241             else:
242                 for g in range(len(x1)):
243                     xs.append(model.x[C[0],x1[g],shelf_i,S_l[ind[shelf_f]],dia])
244     else:
245         return 0
246     return sum(xs)
247 
248 def capacidade_c2(shelf_i,xs1,conf,dia):
249     xs = []
250     ind = ind_listD(shelf_i,xs1)
251     x1 = find_ind_K(1,conf)
252     if len(ind) != 0:
253         for shelf_f in range(len(ind)):
254             if not x1:
255                 return 0
256             else:
257                 for g in range(len(x1)):
258                     xs.append(model.x[C[1],x1[g],shelf_i,S_l[ind[shelf_f]],dia])
259     else:
260         return 0
261     return sum(xs)
262 
263 # introducao das restricoes
264 for dia in range(1, Dbar):
265     for k in range(len(K)):
266         for si in range(len(T_1)):
267             model.Capacidade.add( \
268                 capacidade_c1(T_1[si],T_1, K[k], dia) \
269                     + capacidade_c2(T_1[si],T_1, K[k], dia) \
270                     <= capac_fixa(T_1[si],K[k],Q)*model.y[T_1[si], dia, K[k]])
271 
```

```
272         for si in range(len(T_2)):
273             model.Capacidade.add( \
274                 capacidade_c1(T_2[si],T_2, K[k], dia) \
275                     + capacidade_c2(T_2[si],T_2, K[k], dia) \
276                     <= capac_fixa(T_2[si],K[k],Q)*model.y[T_2[si], dia, K[k]])
277 
278         for si in range(len(T_3)):
279             model.Capacidade.add( \
280                 capacidade_c1(T_3[si],T_3, K[k], dia) \
281                     + capacidade_c2(T_3[si],T_3, K[k], dia) \
282                     <= capac_fixa(T_3[si],K[k],Q)*model.y[T_3[si], dia, K[k]])
283 
284         for si in range(len(T_4)):
285             model.Capacidade.add( \
286                 capacidade_c1(T_4[si],T_4, K[k], dia) \
287                     + capacidade_c2(T_4[si],T_4, K[k], dia) \
288                     <= capac_fixa(T_4[si],K[k],Q)*model.y[T_4[si], dia, K[k]])
289 
290         for si in range(len(T_5)):
291             model.Capacidade.add( \
292                 capacidade_c1(T_5[si],T_5, K[k], dia) \
293                     + capacidade_c2(T_5[si],T_5, K[k], dia) \
294                     <= capac_fixa(T_5[si],K[k],Q)*model.y[T_5[si], dia, K[k]])
295 
296 ## Plantacao
297 model.Plantacao = ConstraintList()
298 
299 def plantacao(shelf, dia):
300     soma = []
301     for i in range(len(shelf)):
302         soma.append(model.x[C[crop],0,'S',shelf[i], dia - Gc[crop] - 1 ])
303     return sum(soma)
304 
305 # introducao das restricoes
306 for crop in range(len(C)):
307     for d in range(Gc[crop]+1,Dbar+1):
308         model.Plantacao.add(plantacao(S,d) == Encomenda[crop][d - 1])
309 
310 ## Configuracao
```

```
311 model.Configuracao = ConstraintList()
312
313 def configuracao(shelf_set,dia,k):
314     soma = []
315     for i in range(len(shelf_set)):
316         for j in range(len(k)):
317             soma.append(model.y[shelf_set[i],dia,k[j]])
318     return sum(soma)
319
320 # introducao das restricoes
321 for dia in range(1,Dbar):
322     for shelf_type in range(len(T)):
323         model.Configuracao.add(configuracao(T[shelf_type],dia,K) \
324                                <= len(T[shelf_type]))
325
326 ## Equilibrio
327 model.Equilibrio = ConstraintList()
328
329 # Membro esquerdo
330 def ind_listE(shelf,xs):
331     ''' Indices das prateleiras com destino shelf
332     >>> ind_listE('s2',T_1)
333     [0,2]
334     as prateleiras com destino s2: 'S' e 's2'
335     '''
336     xe = [] #vazio
337     x1 = []
338     x2 = find_ind_set(xs)
339     x1.append(S_l.index(shelf))
340     if not(shelf in xs):
341         return xe
342     else:
343         for i in range(min(x2)):
344             x1.append(i)
345     x1.sort()
346     return x1
347
348 def rec_el_E(crop,shelf,xs,g,dia):
349     ''' pares corretos com destino shelf '''
```

```
350     x1 = ind_listE(shelf,xs)
351     xs = []
352     for i in range(len(x1)):
353         xs.append(model.x[crop,g,S_l[x1[i]],shelf,dia -1])
354     return sum(xs)
355 
356 # Membro direito
357 def ind_listD(shelf,xs):
358     ''' Indices das prateleiras que saiem de shelf
359     >>> ind_listD('s12',T_4)
360     [12, 15, 16, 17]
361     as prateleiras que saem de 's12':
362         's12','s15','s16','A'
363     '''
364     xe = [] #vazio
365     x1 = []
366     x2 = find_ind_set(xs)
367     x1.append(S_l.index(shelf))
368     if not(shelf in xs):
369         return xe
370     else:
371         for i in range(max(x2)+1,len(S_l)):
372             x1.append(i)
373     x1.sort()
374     return x1
375 
376 def rec_el_D(crop,shelf,xs,g,dia):
377     ''' pares corretos que saem de shelf '''
378     x1 = ind_listD(shelf,xs)
379     xs = []
380     for i in range(len(x1)):
381         xs.append(model.x[crop,g+1,shelf,S_l[x1[i]],dia])
382     return sum(xs)
383 
384 # introducao das restricoes
385 for dia in range(1,Dbar):
386     for crop in range(len(C)):
387         for g in range(Gc[crop]):
388             for s1 in range(len(T_1)):
```

```
                model.Equilibrio.add( rec_el_E(C[crop],T_1[s1],T_1, g, dia) \
                                    == rec_el_D(C[crop],T_1[s1], T_1, g, dia) )

            for s2 in range(len(T_2)):
                model.Equilibrio.add( rec_el_E(C[crop],T_2[s2],T_2, g, dia) \
                                    == rec_el_D(C[crop],T_2[s2], T_2, g, dia) )

            for s3 in range(len(T_3)):
                model.Equilibrio.add( rec_el_E(C[crop],T_3[s3],T_3, g, dia) \
                                    == rec_el_D(C[crop],T_3[s3], T_3, g, dia) )

            for s4 in range(len(T_4)):
                model.Equilibrio.add( rec_el_E(C[crop],T_4[s4],T_4, g, dia) \
                                    == rec_el_D(C[crop],T_4[s4], T_4, g, dia) )

            for s5 in range(len(T_5)):
                model.Equilibrio.add( rec_el_E(C[crop],T_5[s5],T_5, g, dia) \
                                    == rec_el_D(C[crop],T_5[s5], T_5, g, dia) )

### Fixacao de Variaveis
## FV1
model.FV_1 = ConstraintList()

for crop in range(len(C)):
    for dia in range(Dbar):
        for g in range(Gc[crop]+1):
            model.FV_1.add(model.x[C[crop],g,'S','A', dia] == 0)
            model.FV_1.add(model.x[C[crop],g,'S','S', dia] == 0)
            model.FV_1.add(model.x[C[crop],g,'A','A', dia] == 0)

## FV2
model.FV_2 = ConstraintList()

for dia in range(Dbar):
    for crop in range(len(C)):
        for g in range(1,Gc[crop]+1):
            for shelf in range(len(S_1)):
                model.FV_2.add(model.x[C[crop],g, 'S', S_1[shelf], dia] == 0)
```

```
        for g in range(Gc[crop]):
            for shelf in range(len(S_l)):
                model.FV_2.add(model.x[C[crop],g, S_l[shelf], 'A', dia] == 0)

for dia in range(1,Dbar):
    for crop in range(len(C)):
        for g in [0,Gc[crop]]:
            for s1 in range(len(T_1)):
                for s2 in range(len(T_2)):
                    model.FV_2.add(model.x[C[crop],g,T_1[s1],T_2[s2],dia] == 0)

                for s3 in range(len(T_3)):
                    model.FV_2.add(model.x[C[crop],g,T_1[s1],T_3[s3],dia] == 0)

                for s4 in range(len(T_4)):
                    model.FV_2.add(model.x[C[crop],g,T_1[s1],T_4[s4],dia] == 0)

                for s5 in range(len(T_5)):
                    model.FV_2.add(model.x[C[crop],g,T_1[s1],T_5[s5],dia] == 0)

            for s2 in range(len(T_2)):
                for s3 in range(len(T_3)):
                    model.FV_2.add(model.x[C[crop],g,T_2[s2],T_3[s3],dia] == 0)

                for s4 in range(len(T_4)):
                    model.FV_2.add(model.x[C[crop],g,T_2[s2],T_4[s4],dia] == 0)

                for s5 in range(len(T_5)):
                    model.FV_2.add(model.x[C[crop],g,T_2[s2],T_5[s5],dia] == 0)

            for s3 in range(len(T_3)):
                for s4 in range(len(T_4)):
                    model.FV_2.add(model.x[C[crop],g,T_3[s3],T_4[s4],dia] == 0)

                for s5 in range(len(T_5)):
                    model.FV_2.add(model.x[C[crop],g,T_3[s3],T_5[s5],dia] == 0)

            for s4 in range(len(T_4)):
                for s5 in range(len(T_5)):
```

```
                    model.FV_2.add(model.x[C[crop],g,T_4[s4],T_5[s5],dia] == 0)

            for shelf in range(len(S)):
                model.FV_2.add(model.x[C[crop],g,S[shelf],S[shelf],dia] == 0)

## FV3
model.FV_3 = ConstraintList()

for dia in range(Dbar):
    for crop in range(len(C)):
        for sh1 in range(len(T_1)):
            for sh2 in range(1+len(T_1)+1,len(S_l)):
                model.FV_3.add(model.x[C[crop],0,T_1[sh1],S_l[sh2],dia] == 0)

        for sh2 in range(len(T_2)):
            for sh3 in range(1+len(T_1)+len(T_2)+1,len(S_l)):
                model.FV_3.add(model.x[C[crop],0,T_2[sh2],S_l[sh3],dia] == 0)

        for sh3 in range(len(T_3)):
            for sh4 in range(1+len(T_1)+len(T_2)+len(T_3)+1,len(S_l)):
                model.FV_3.add(model.x[C[crop],0,T_3[sh3],S_l[sh4],dia] == 0)

        for sh4 in range(len(T_4)):
            for sh5 in range(1+len(T_1)+len(T_2)+len(T_3)+len(T_4)+1,len(S_l)):
                model.FV_3.add(model.x[C[crop],0,T_4[sh4],S_l[sh5],dia] == 0)

        for sh5 in range(len(T_5)):
            model.FV_3.add(model.x[C[crop],0, T_5[sh5], 'A', dia] == 0)

        for shelf in range(len(S)):
            model.FV_3.add(model.x[C[crop],0, S[shelf], S[shelf],dia] == 0)

for dia in range(Dbar):
    for crop in range(len(C)):
        for sh in range(1,len(S_l)-1):
            model.FV_3.add(model.x[C[crop],Gc[crop],'S',S_l[sh],dia] == 0)

        for sh1 in range(len(T_1)):
            for sh2 in range(1+len(T_1)+1,len(S_l)-1):
```

```
506                 model.FV_3.add(model.x[C[crop],Gc[crop],T_1[sh1],S_l[sh2],dia] == 0)
507
508         for sh2 in range(len(T_2)):
509             for sh3 in range(1+len(T_1)+len(T_2)+1,len(S_l)-1):
510                 model.FV_3.add(model.x[C[crop],Gc[crop],T_2[sh2],S_l[sh3],dia] == 0)
511
512         for sh3 in range(len(T_3)):
513             for sh4 in range(1+len(T_1)+len(T_2)+len(T_3)+1,len(S_l)-1):
514                 model.FV_3.add(model.x[C[crop],Gc[crop],T_3[sh3],S_l[sh4],dia] == 0)
515
516         for sh4 in range(len(T_4)):
517             for sh5 in range(1+len(T_1)+len(T_2)+len(T_3)+len(T_4)+1,len(S_l)-1):
518                 model.FV_3.add(model.x[C[crop],Gc[crop],T_4[sh4],S_l[sh5],dia] == 0)
519
520         for shelf in range(len(S)):
521             model.FV_3.add(model.x[C[crop],Gc[crop],S[shelf],S[shelf],dia] == 0)
522
523 ## FV4
524 model.FV_4 = ConstraintList()
525
526 def ultimo_dia_procura(lista):
527     xs = []
528     for i in range(len(lista)):
529         if lista[i]!=0:
530             xs.append(i)
531         else:
532             continue
533     return max(xs)+1
534
535 def ult_dia_proc(crop,lista):
536     dia_l = ultimo_dia_procura(lista[crop])
537     ult_dia_poss = dia_l - Gc[crop] + 1
538     return [dia_l,ult_dia_poss]
539
540 for crop in range(len(C)):
541     xs = ult_dia_proc(crop,Encomenda)
542     for dia in range(xs[1], xs[0] + 1):
543         for g in range(Gc[crop] - (xs[0] - dia) +1):
544
```

```
            for sh1 in range(len(T_1)):
                for sh2 in range(len(T_2)):
                    model.FV_4.add(model.x[C[crop],g,T_1[sh1],T_2[sh2],dia] == 0)

                for sh3 in range(len(T_3)):
                    model.FV_4.add(model.x[C[crop],g,T_1[sh1],T_3[sh3],dia] == 0)

                for sh4 in range(len(T_4)):
                    model.FV_4.add(model.x[C[crop],g,T_1[sh1],T_4[sh4],dia] == 0)

                for sh5 in range(len(T_5)):
                    model.FV_4.add(model.x[C[crop],g,T_1[sh1],T_5[sh5],dia] == 0)

            for sh2 in range(len(T_2)):
                for sh3 in range(len(T_3)):
                    model.FV_4.add(model.x[C[crop],g,T_2[sh2],T_3[sh3],dia] == 0)

                for sh4 in range(len(T_4)):
                    model.FV_4.add(model.x[C[crop],g,T_2[sh2],T_4[sh4],dia] == 0)

                for sh5 in range(len(T_5)):
                    model.FV_4.add(model.x[C[crop],g,T_2[sh2],T_5[sh5],dia] == 0)

            for sh3 in range(len(T_3)):
                for sh4 in range(len(T_4)):
                    model.FV_4.add(model.x[C[crop],g,T_3[sh3],T_4[sh4],dia] == 0)

                for sh5 in range(len(T_5)):
                    model.FV_4.add(model.x[C[crop],g,T_3[sh3],T_5[sh5],dia] == 0)

            for sh4 in range(len(T_4)):
                for sh5 in range(len(T_5)):
                    model.FV_4.add(model.x[C[crop],g,T_4[sh4],T_5[sh5],dia] == 0)

            for shelf in range(len(S)):
                model.FV_4.add(model.x[C[crop],g, S[shelf], S[shelf],dia] == 0)

for crop in range(len(C)):
    for shelf in range(len(S)):
```

```
584         model.FV_4.add(model.x[C[crop],0,'S',S[shelf],Dbar-1] == 0)
585
586 ## FV5
587 model.FV_5 = ConstraintList()
588
589 for dia in range(Dbar):
590     for crop in range(len(C)):
591         if dia < Gc[crop]:
592             for g in range(dia+1, Gc[crop]+1):
593                 for si in range(len(S_1)):
594                     model.FV_5.add(model.x[C[crop],g, 'S', S_1[si], dia] == 0)
595                     model.FV_5.add(model.x[C[crop],g, S_1[si], 'A', dia] == 0)
596
597                 for sh1 in range(len(T_1)):
598                     for sh2 in range(len(T_2)):
599                         model.FV_5.add(model.x[C[crop],g,T_1[sh1],T_2[sh2],dia] == 0)
600
601                     for sh3 in range(len(T_3)):
602                         model.FV_5.add(model.x[C[crop],g,T_1[sh1],T_3[sh3],dia] == 0)
603
604                     for sh4 in range(len(T_4)):
605                         model.FV_5.add(model.x[C[crop],g,T_1[sh1],T_4[sh4],dia] == 0)
606
607                     for sh5 in range(len(T_5)):
608                         model.FV_5.add(model.x[C[crop],g,T_1[sh1],T_5[sh5],dia] == 0)
609
610                 for sh2 in range(len(T_2)):
611                     for sh3 in range(len(T_3)):
612                         model.FV_5.add(model.x[C[crop],g,T_2[sh2],T_3[sh3],dia] == 0)
613
614                     for sh4 in range(len(T_4)):
615                         model.FV_5.add(model.x[C[crop],g,T_2[sh2],T_4[sh4],dia] == 0)
616
617                     for sh5 in range(len(T_5)):
618                         model.FV_5.add(model.x[C[crop],g,T_2[sh2],T_5[sh5],dia] == 0)
619
620                 for sh3 in range(len(T_3)):
621                     for sh4 in range(len(T_4)):
622                         model.FV_5.add(model.x[C[crop],g,T_3[sh3],T_4[sh4],dia] == 0)
```

```
623
624                     for sh5 in range(len(T_5)):
625                         model.FV_5.add(model.x[C[crop],g,T_3[sh3],T_5[sh5],dia] == 0)
626
627             for sh4 in range(len(T_4)):
628                 for sh5 in range(len(T_5)):
629                     model.FV_5.add(model.x[C[crop],g,T_4[sh4],T_5[sh5],dia] == 0)
630
631             for shelf in range(len(S)):
632                 model.FV_4.add(model.x[C[crop],g, S[shelf], S[shelf],dia] == 0)
633         else: break
634
635 ## FV6
636 model.FV_6 = ConstraintList()
637
638 for crop in range(len(C)):
639     for dia in range(Gc[crop]+1, Dbar+1):
640         for shelf in range(len(S)):
641             model.FV_6.add( \
642                     model.x[C[crop],0, 'S', S[shelf], dia-Gc[crop]-1] \
643                           <= Encomenda[crop][dia-1])
644
645 for crop in range(len(C)):
646     for dia in range(1,Dbar+1):
647         for shelf in range(len(S)):
648             model.FV_6.add( \
649                     model.x[C[crop],Gc[crop], S[shelf], 'A', dia-1] \
650                           <= Encomenda[crop][dia-1])
651
652 ## FV7
653 model.FV_7 = ConstraintList()
654
655 def soma_procura(crop,dia, k):
656     x1 = []
657     for dl in range(dia +1, dia + Gc[crop]+1):
658         if dl <= len(Encomenda[crop]):
659             dk = Gc[crop] + dia +1 - dl
660             if Kg[crop][dk-1] == k:
661                 x1.append(Encomenda[crop][dl-1])
```

```
662             else:
663                 continue
664         else:
665             break
666     if x1==[]:
667         x1.append(0)
668     return x1
669 
670 def eta_dk(crop):
671     ''' calcula eta_dk para a crop c '''
672     soma=[]
673     for dia in range(1,Dbar):
674         soma_I1 = soma_procura(crop,dia,'I1')
675         soma_I2 = soma_procura(crop,dia,'I2')
676         soma_I3 = soma_procura(crop,dia,'I3')
677         soma_I4 = soma_procura(crop,dia,'I4')
678         soma_I5 = soma_procura(crop,dia,'I5')
679         soma.append([sum(soma_I1),sum(soma_I2),\
680                      sum(soma_I3),sum(soma_I4),sum(soma_I5)])
681     return soma
682 
683 def eta_dk_P():
684     ''' calcula o valor total de eta_dk '''
685     xsm = []
686     for d in range(Dbar-1):
687         i1 = sum(eta_dk(i)[d][0] for i in range(len(C)))
688         i2 = sum(eta_dk(i)[d][1] for i in range(len(C)))
689         i3 = sum(eta_dk(i)[d][2] for i in range(len(C)))
690         i4 = sum(eta_dk(i)[d][3] for i in range(len(C)))
691         i5 = sum(eta_dk(i)[d][4] for i in range(len(C)))
692         xsm.append([i1,i2,i3,i4,i5])
693     return xsm
694 
695 for dia in range(1,Dbar):
696     for k in range(len(K)):
697         if eta_dk_P()[dia-1][k]==0:
698             for shelf in range(len(S)):
699                 model.FV_7.add(model.y[S[shelf],dia,K[k]]==0)
700         else: continue
```

```
701
702 ### Solucao
703 opt=SolverFactory('glpk')
704 solucao = opt.solve(model,tee=True)
705 model.Objetivo.display() # mostrar o valor do objetivo
706
707 # aceder aos valores das variaveis x e y
708 for i in model.x:
709     if model.x[i].value != 0 :
710         print(str(model.x[i]), model.x[i].value)
711
712 for i in model.y:
713     if model.y[i].value != 0:
714         print(str(model.y[i]), model.y[i].value)
```