RELATÓRIO
Modelo de Otimização para Planejamento e Controle da Produção: Estudo de Caso para uma Empresa de Ampolas de Vidro para Garrafas Térmicas

Larissa Moreira, Milena Nobres, Rômulo Monteiro

[1]Universidade Federal de Viçosa - *Campus* Rio Paranaíba
Rodovia BR 230 KM 7, Rio Paranaíba - MG, 38810-000

1. Descrição do Problema

As embalagens são fundamentais para proteger e conservar produtos em geral, podendo ser feitas de diversos materiais como plástico, vidro, papel ou metal. O tipo de embalagem que mais se destaca positivamente é a de vidro, pois é o material que menos interfere no sabor, odor e qualidade do produto. Além disso, embalagens de vidro são práticas no sentido de que são resistentes a diferentes temperaturas, podendo ser levadas à mesa ou ao micro-ondas, e também são menos prejudiciais ao meio ambiente.

O principal problema a ser otimizado neste modelo matemático é o planejamento e o controle da produção de ampolas térmicas em uma fábrica especializada neste ramo. Nesta empresa são fabricadas cinco tipos de ampolas que variam de acordo com: (i) peso, (ii) custo, (iii) estoque de segurança, (iv) refugo (desperdício diário), (v) número de peças extraídas por máquina, (vi) estoque no início do período e (vii) demanda empurrada do produto, ou seja, em que há uma incerteza.

Na fábrica em que as ampolas são produzidas, existe um único forno regenerativo de porta traseira com capacidade de derretimento limitado, o qual alimenta duas máquinas que possuem a mesma configuração de quatro seções e boca dupla. Estas máquinas trabalham em conjunto, e uma produz a parte interna enquanto a outra é responsável pela produção da parte externa das ampolas. Pode-se observar na Figura 1 que ocorrem quedas na produção em alguns dias com quatro tipos de ampolas, onde é indicado por uma seta. Isso ocorre devido os tempos de **set-up**, que é o tempo gasto na preparação da máquina para produzir outro tipo de ampola.

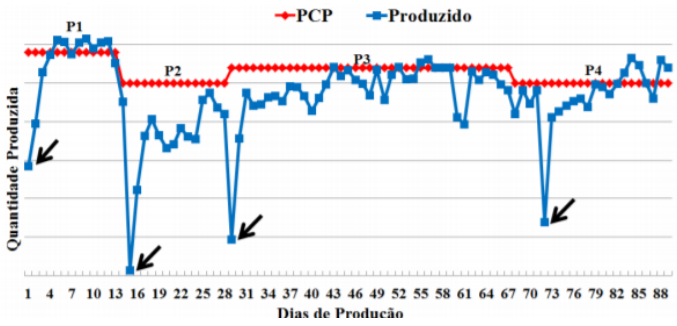


Figura 1. Produção realizada em um trimestre

Na Figura 2 é mostrado o refugo, que é a taxa de desperdício do produto, e a eficiência, que é a taxa de produtos sem defeitos que foram aprovados no controle. Ambas as taxas são calculadas por dia.

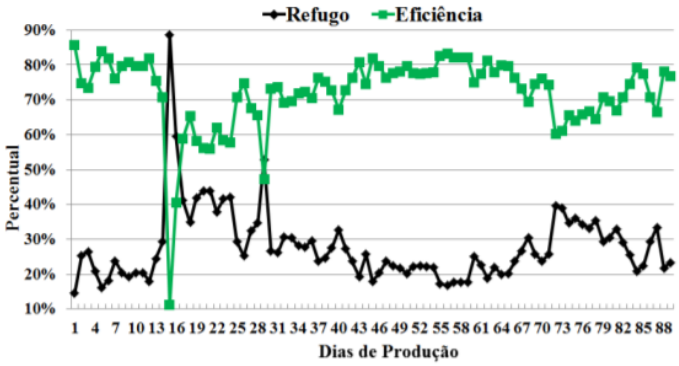


Figura 2. Refugo e eficiência em um trimestre

Na Figura 3 é apresentado o **ramp-up**, que é uma influência maior do refugo agindo na produção depois dos tempos de set-up, ou seja, é o tempo de adaptação da nova linha de produção e é marcada por um crescimento gradual da produção até alcançar sua estabilização. Durante o período de ramp-up, existe uma queda na produção durante alguns dias e isso acontece devido os tempos de set-up entre produtos e período de ramp-up.

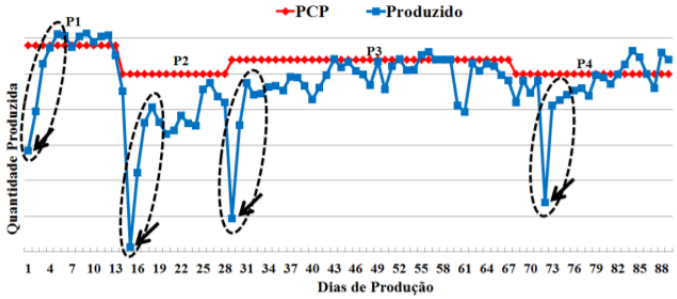


Figura 3. Comportamento do período de ramp-up durante a produção real

2. Modelo Matemático Implementado e Restrições

Nesta sessão serão apresentados a função objetivo e as restrições e como foram implementadas.

2.1. Função Objetivo

(1)

$$\min \sum_{i \in I} \sum_{t \in T} \sum_{s \in S} \mathbf{W}its * \mathbf{QE}i * (1 - \mathbf{TRU}is) * \mathbf{C}i + 0,01 * \mathbf{EU} * sum_{i \in I} \mathbf{EstS}i * \mathbf{C}i$$

Primeira parte da FO: minimiza os custos de tempo de set-Up e ramp-up, pois a demora na troca de linha de produção entre os diferentes tipos de ampolas traz custos para a indústria então, então se os tempos de pausa e de produção forem diminuídos a empresa terá menos gastos.

Segunda parte da FO: é uma fórmula matemática que vai impedir a produção exagerada de uma determinada ampola depois que ela atingir o estoque de segurança.

```
// DECLARAÇÃO DA FUNÇÃO OBJETIVO
//(1)
IloExpr fo(env);
IloExpr obj(env);
IloNum x;
x = 1;
for (int a = 0; a < A; a++) {
    for (int t = 0; t < T; t++) {
        for (int s= 0; s < S; s++) {
            obj += Set[a][t][s] * QuantidadeE[a] * (x - TRamp[a][s]) * Custo[a];
            fo += 0.001 * U * SEstoque[a] * Custo[a];
        }
    }
}
fo = fo + obj;
```

Figura 4. Função objetivo

2.2. Restrições de Balanceamento de estoque

(2) Estit = **ProdL**it - **Dem**it + **Est**i(t - 1) $\forall(i, t > 1)$:

Para tempo maior que 1, o estoque se dá pela produção líquida subtraído a demanda e somado ao estoque do tempo ***t*** anterior, no qual para todo produto ***i***. Ao passar essa restrição para o código foi necessário fazer o *for* começar de 1 para que não acesse uma posição negativa, mesmo com a comparação do *if*.

(3) Estit = **ProdL**it - **Dem**it + **EstI**i $\forall(i, t = 1)$:

Para tempo igual a 1, o estoque se dá pela produção líquida subtraída a demanda e somada ao estoque inicial, pois não houve estoque anterior.

```
// (2)&(3)Restrição Balanceamento de estoque
if (T > 1) {
    for (int a = 1; a < A; a++) {
        for (int t = 1; t < T; t++) {
            IloExpr r2(env);
            r2 = ProdL[a][t] - Demanda[a][t] + Estoque[a][t-x];
            IloConstraint Balanc = Estoque[a][t] == r2;
            stringstream var;
            var << "Balanceamento[" << a << "][" << t << "]";
            Balanc.setName(var.str().c_str());
            modelo.add( Balanc );
        }
    }
}
//(3)
else if (T == 1) {
    for (int a = 0; a < A; a++) {
        for (int t = 0; t < T; t++) {
            IloExpr r3(env);
            r3 = ProdL[a][t] - Demanda[a][t] + IEstoque[a];
            IloConstraint Balanc1 = Estoque[a][t] == r3;
            stringstream var;
            var << "Balanceamento[" << a << "][" << t << "]";
            Balanc1.setName(var.str().c_str());
            modelo.add(Balanc1);
        }
    }
}
```

Figura 5. Restrição 2 e 3

2.3. Restrição de equilíbrio de estoque de segurança

(4) Estit / **EstS**i $\geq$ **EU**

Estoque do produto i em um determinado tempo t dividido pelo estoque de segurança do produto i, e o resultado dessa divisão tem quer ser menor ou igual ao número de vezes que o estoque de segurança foi ultrapassado.

```
// (4)Restrição equilibrio de estoque de segurança
for (int a = 0; a < A; a++ ) {
    for (int t = 0; t < T; t++) {
        IloExpr r4(env);
        r4 = Estoque[a][t] / SEstoque[a];
        IloConstraint Equi = U >= r4;
        stringstream var;
        var << "Equilibrio[" << a << "][" << t << "]";
        Equi.setName(var.str().c_str());
        modelo.add(Equi);

    }
}
```

Figura 6. Restrição 4

2.4. Restrições de linearização

(5) $\mathbf{V}\text{its} \geq \mathbf{ProdT}it - \mathbf{B} \cdot (1 - \mathbf{W}its)$

```
//(5)(6)(7)(8) Linearização
//(5)
for (int a = 0; a < A; a++) {
    for (int t = 0; t < T; t++) {
        for (int s = 0; s < S ; s++) {
            IloExpr r5(env);
            r5 = ProdT[a][t] - 1000000 * (x - Set[a][t][s]);
            IloConstraint Linear1 = Linear[a][t][s] >= r5;
            stringstream var;
            var << "Linear1[" << a << "][" << t << "][" << s << "]";
            Linear1.setName(var.str().c_str());
            modelo.add(Linear1);
        }
    }
}
```

Figura 7. Restrição 5

(6) $\mathbf{V}\text{its} \leq \mathbf{ProdT}it + \mathbf{B} \cdot (1 - \mathbf{W}its)$

```
//(6)
for (int a = 0; a < A; a++) {
    for (int t = 0; t < T; t++) {
        for (int s = 0; s < S; s++) {
            IloExpr r6(env);
            r6 = ProdT[a][t] + 1000000 * (1 - Set[a][t][s]);
            IloConstraint Linear2 = Linear[a][t][s] <= r6;
            stringstream var;
            var << "Linear2[" << a << "][" << t << "][" << s << "]";
            Linear2.setName(var.str().c_str());
            modelo.add(Linear2);
        }
    }
}
```

Figura 8. Restrição 6

(7) $\mathbf{W}\text{its} \cdot \mathbf{B} \geq \mathbf{V}its$

(8) $\mathbf{W}\text{its} \cdot \mathbf{B} \leq \mathbf{V}its$

Em períodos de ramp-up há um desequilíbrio na produção, onde essas restrições vêm para controlá-lo e voltar a produção linear.

```
//(7)
for (int a = 0; a < A; a++) {
    for (int t = 0; t < T; t++) {
        for (int s = 0; s < S; s++) {
            IloExpr r7(env);
            r7 = 1000000 * Set[a][t][s];
            IloConstraint Linear3 = Linear[a][t][s] <= r7;
            stringstream var;
            var << "Linear3[" << a << "][" << t << "][" << s << "]";
            Linear3.setName(var.str().c_str());
            modelo.add(Linear3);
        }
    }
}
```

Figura 9. Restrição 7

```
//(8)
for (int a = 0; a < A; a++) {
    for (int t = 0; t < T; t++) {
        for (int s = 0; s < S; s++) {
            IloExpr r8(env);
            r8 = (-Set[a][t][s]) * 1000000 ;
            stringstream var;
            IloConstraint Linear4 = Linear[a][t][s] >= r8;
            var << "Linear4[" << a << "][" << t << "]["<< s << "]";
            Linear4.setName(var.str().c_str());
            modelo.add(Linear4);
        }
    }
}
```

Figura 10. Restrição 8

2.5. Restrição de produção líquida

(9 ProdLit = $\sum_{s \in S}$ **V**$its \cdot$ **TRU**$is + ($**ProdT**$it - \sum_{s \in S}$ **V**$its) \cdot (1 -$ **R**$it) \forall (i, t)$

Se houver set-up, então ocorre influência do ramp-up na produção onde a variável auxiliar para a linearização é multiplicada pela taxa de ramp-up, que é a primeira parte da função. Caso não houver tempos de set-up, a produção líquida se dá pelo produto total, menos a variável auxiliar de linearização vezes o refugo do produto.

```
//(9)Restrição Produção Liquida
for (int a = 0; a < A; a++) {
    for (int t = 0; t < T; t++) {
        IloExpr PD(env);
        IloExpr PD1(env);
       // IloExpr PD2(env);
        for (int s = 0; s < S; s++) {
            PD1 += Linear[a][t][s] * TRamp[a][s];
            PD += (ProdT[a][t] - Linear[a][t][s]) * (x - Refulgo[a][t]);
            PD = PD1 + PD;
            IloConstraint ProdLi = PD == ProdL[a][t];
            stringstream var;
            var << "ProdLiquida[" << a << "][" << t << "][" << s << "]";
            ProdLi.setName(var.str().c_str());
            modelo.add(ProdLi);
        }
    }
}
```

Figura 11. Restrição 9

2.6. Restrição de Produção Total

(10) ProdLit = **QE**i $\cdot$**X**$it \forall (i, t)$

A produção líquida é igual a quantidade extraída de ampola (em dias normais sem set-up), se houver ou não produção de produto em determinado tempo.

```
//(10)Restrição Produto Total
for (int a = 0; a < A; a++) {
    for (int t = 0; t < T; t++) {
        IloExpr r10(env);
        r10 = QuantidadeE[a] * PP[a][t];
        IloConstraint ProdTt = r10 == ProdT[a][t];
        stringstream var;
        var << "ProdTotal[" << a << "][" << t << "]";
        ProdTt.setName(var.str().c_str());
         modelo.add(ProdTt);
    }
}
```

Figura 12. 10

2.7. Restrição somente um produto por vez

(11) $\sum_{i \in I} \mathbf{X}it = 1 \forall (i,t)$

Em um determinado tempo as máquinas podem trabalhar somente com um tipo de ampola.

Xit: Produção do produto i no período de tempo t

```
//(11)Restrição um produto por maquina
for (int t = 0; t < T; t++) {
    IloExpr r11(env);
    for (int a = 0; a < A; a++) {
        r11 += PP[a][t];
        IloConstraint Prod1 = r11 == 1;
        stringstream var;
        var << "UmProduto[" << a << "][" << t << "]";
        Prod1.setName(var.str().c_str());
        modelo.add(Prod1);
    }
}
```

Figura 13. Restrição 11

2.8. Restrições de tempo de set-up

(12) **S**it $\geq \mathbf{X}it - \mathbf{X}i(t-1) \forall (i, t > 1)$

Para tempo maior que 1, a produção no tempo atual menos a produção no tempo anterior deve que ser menor que a variável de set-up

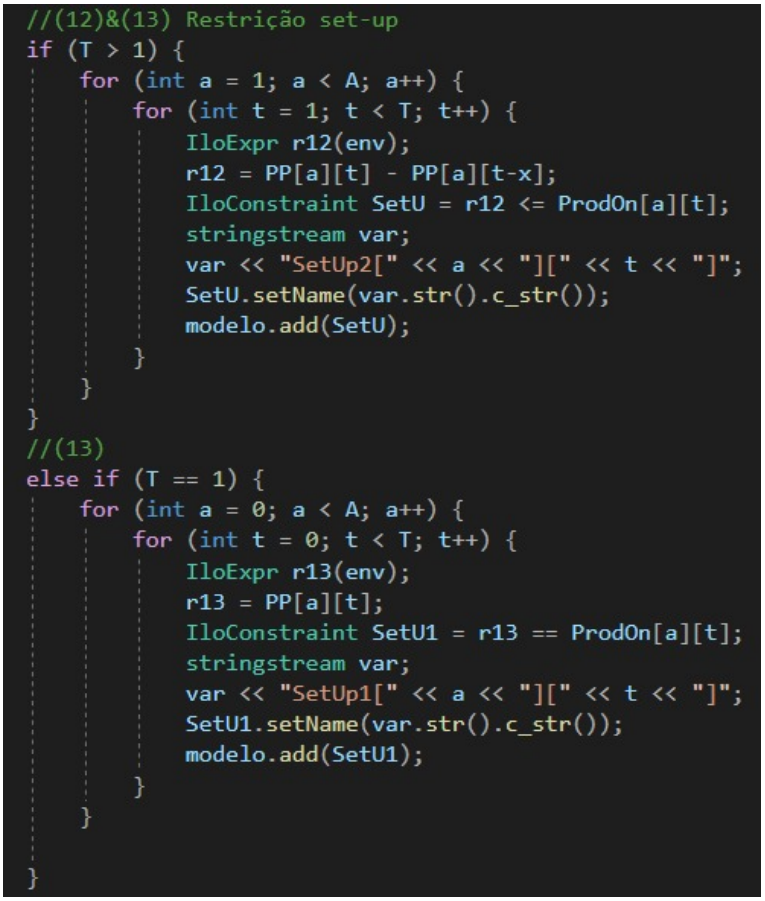
Sit: É uma variável binária que indica se houve ou não set-up na produção do produto i no período de tempo t

(13) **S**it = **X**it

Quando o tempo for igual a 1, a variável binária de set-up deve ser igual a produção do produto.

2.9. Restrição de estágios de ramp-up

(14) **S**it = **W**i(t+s)s $\forall (i, s, t < T - s)$



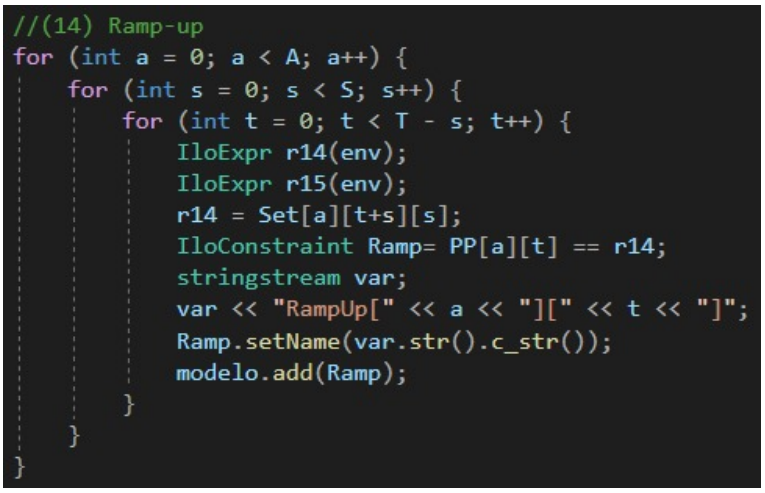
```
//(12)&(13) Restrição set-up
if (T > 1) {
    for (int a = 1; a < A; a++) {
        for (int t = 1; t < T; t++) {
            IloExpr r12(env);
            r12 = PP[a][t] - PP[a][t-x];
            IloConstraint SetU = r12 <= ProdOn[a][t];
            stringstream var;
            var << "SetUp2[" << a << "][" << t << "]";
            SetU.setName(var.str().c_str());
            modelo.add(SetU);
        }
    }
}
//(13)
else if (T == 1) {
    for (int a = 0; a < A; a++) {
        for (int t = 0; t < T; t++) {
            IloExpr r13(env);
            r13 = PP[a][t];
            IloConstraint SetU1 = r13 == ProdOn[a][t];
            stringstream var;
            var << "SetUp1[" << a << "][" << t << "]";
            SetU1.setName(var.str().c_str());
            modelo.add(SetU1);
        }
    }

}
```

Figura 14. Restrições 12 e 13

Quando houver um set-up, a variável de ramp-up é ativada.

Wits: Indica se produto i, no período de tempo t e no dia s, está no período de ramp-up. Variável binária que é ativada 3 dias após o set-up.



```
//(14) Ramp-up
for (int a = 0; a < A; a++) {
    for (int s = 0; s < S; s++) {
        for (int t = 0; t < T - s; t++) {
            IloExpr r14(env);
            IloExpr r15(env);
            r14 = Set[a][t+s][s];
            IloConstraint Ramp= PP[a][t] == r14;
            stringstream var;
            var << "RampUp[" << a << "][" << t << "]";
            Ramp.setName(var.str().c_str());
            modelo.add(Ramp);
        }
    }
}
```

Figura 15. Restrição 14

2.10. Restrições de Domínio

(15) Estit, **ProdL**it, **ProdT**it, **V**its, **EU** ≥ 0

(16)Xit, **S**it, **W**$it \in \{0, 1\}$

Restrições de negatividade e de variáveis binárias, que são implementadas na declaração das variáveis.

3. Descrição das Instâncias

- **I** : Tipos de ampolas a serem fabricadas
- **S**: Dias de ramp-up
- **T**: Período de Tempo
- **C*i***: Custo do produto ***i***
- **Dem*it***: Demanda do produto ***i*** no período de tempo ***t***
- **EstS*i***: Estoque de Segurança do produto ***i***
- **EstI*i***: Estoque Inicial do produto ***i***
- **TRU*is***: Taxa de ramp-up do produto ***i*** no dia ***s***
- **QE*i***: Quantidade de extração do produto ***i*** (em dia normal sem set-up)
- **R*it***: Refugo do produto ***i*** no período de tempo ***t***

4. Interpretação dos Resultados e Comparação com o Artigo Escolhido

Ao executar o programa, o mesmo não consegue achar a solução ótima do problema, que pode ser causada por declarações de instância, ou a implementação de algumas restrições como a restrição (2) que se refere ao balanceamento de estoque, restrição (9) Produção Líquida, e a restrição (12) Set-up. Assim, sem a solução ótima, não foi possível fazer a comparação entre os resultados obtidos no desenvolvimento deste projeto com o artigo base.

5. Dificuldades Encontradas no Desenvolvimento do Trabalho

Ao realizar este trabalho, enfrentamos dificuldades ao declarar variáveis de três dimensões; ao tentar acessar um índice que é maior do que o tamanho do vetor; ao implementar as restrições 2, 9, 12 e 14 ; ao tentar resolver o estouro aritmético; ao tentar descobrir o motivo de não estar encontrando o Ótimo ao executar o código.

6. Referências

Amorim, Flaviana et al. Modelo de Otimização para Planejamneto e Controle da Produção: Estudo de Caso para uma Empresa de Ampolas de Vidro para Garrafas Térmicas. In: **ANAIS DO LII SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL**, 2020, João Pessoa. Anais eletrônicos... Campinas, Galoá, 2020. Disponível em: ¡https://proceedings.science/sbpo-2020/papers/modelo-de-otimizacao-para-planejamento-e-controle-da-producao–estudo-de-caso-para-uma-empresa-de-ampolas-de-vidro-para-¿ Acesso em: 15 Maio. 2021.