自社にとって最適な生産計画の作り方セミナー

～ 120分で理解する、生産計画自動生成ツールの活用方法 ～

14:00 開始予定

マイク、カメラをOFFにしてしばらくお待ちください

Zoomの表示名は、セミナー申し込み時の
お名前としていただけますようご協力お願いいたします

FIXSTARS Amplify

FIXSTARS Amplify

自社にとって最適な生産計画の作り方セミナー

～ 120分で理解する、生産計画自動生成ツールの活用方法！ ～

本日のAgenda

- はじめに
- 会社紹介
- Fixstars Amplify Scheduling Engine のご紹介
- 生産計画最適化のワークショップ

休憩

- 生産計画最適化のワークショップ (続き)
- 事例、価格、今後の進め方等のご紹介
- Wrap Up

質問は随時ZoomのチャットかQ&Aでお願いします

はじめに

本セミナーのゴール

- 最適化問題を解くためのクラウドサービス「Fixstars Amplify」と、その一製品である「Fixstars Amplify Scheduling Engine」(Amplify SE) を知り、Amplify SE を使ってスケジューリング最適化問題を解くための手法やプロセスを理解する

- ワークショップを通して、実際に Amplify SE を使ってみることで、自社の業務への適用のイメージを掴む

質問は随時ZoomのチャットかQ&Aでお願いします

会社紹介

フィックスターズの基本情報

会社名	株式会社フィックスターズ
本社所在地	東京都港区芝浦3-1-1 msb Tamachi 田町ステーションタワーN 28階
設立	2002年8月
上場区分	東証プライム（証券コード：3687）
代表取締役社長	三木　聡

資本金	5億5,446万円
社員数（連結）	263名（2022年9月現在）
主なお客様	キオクシア株式会社 ルネサスエレクトロニクス株式会社 トヨタグループ（トヨタ自動車株式会社・ 豊田通商株式会社・株式会社デンソー） みずほ証券株式会社 キヤノン株式会社

グループ会社

2021/10/1～
（フィックスターズからスピンアウト）

Fixstars Solutions, Inc.
完全子会社
米国での営業及び開発を担当

(株)Fixstars Autonomous Technologies
株式会社ネクスティ エレクトロニクスとのJV
自動運転向けソフトウェアを開発

(株)Fixstars Amplify
完全子会社
量子コンピューティングのクラウド事業を運営

(株)Sider
完全子会社
開発支援SaaS「Sider」を運営

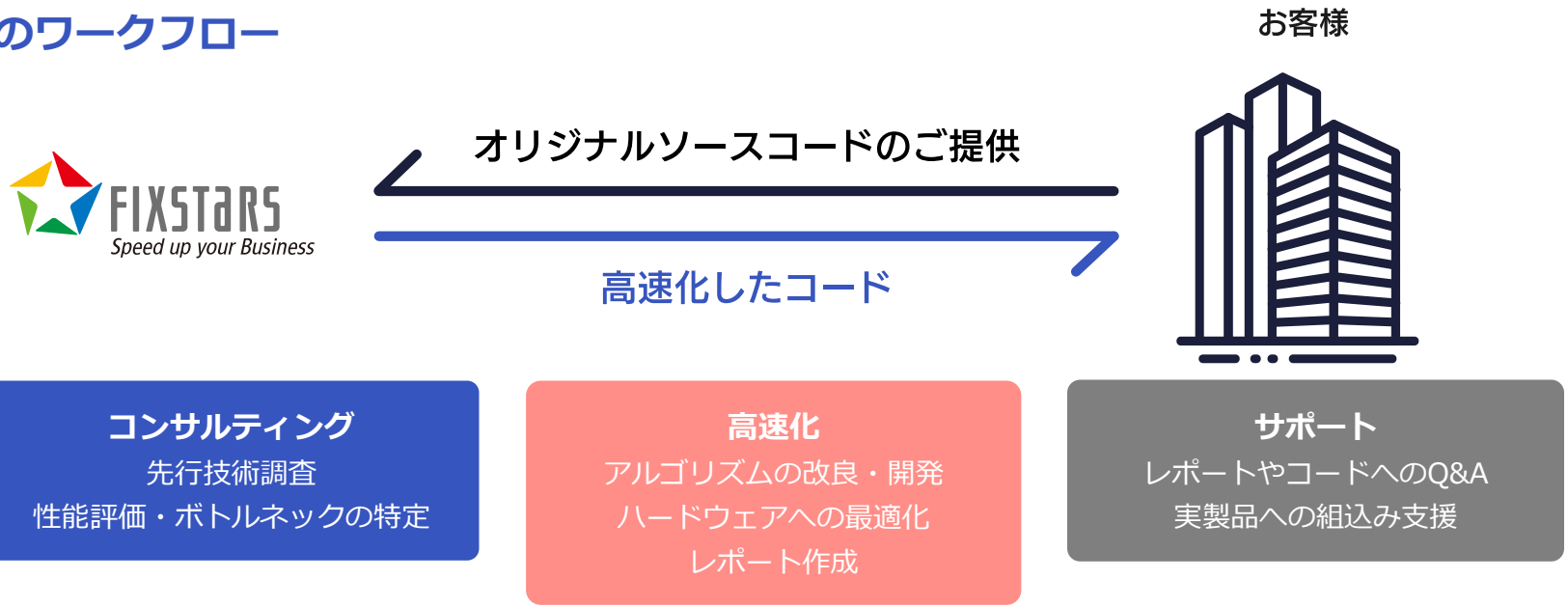
(株)Smart Opinion
連結子会社
乳がんAI画像診断支援事業を運営

オスカーテクノロジー(株)
連結子会社
ソフトウェア自動並列化サービスを提供

フィックスターズのサービス概要

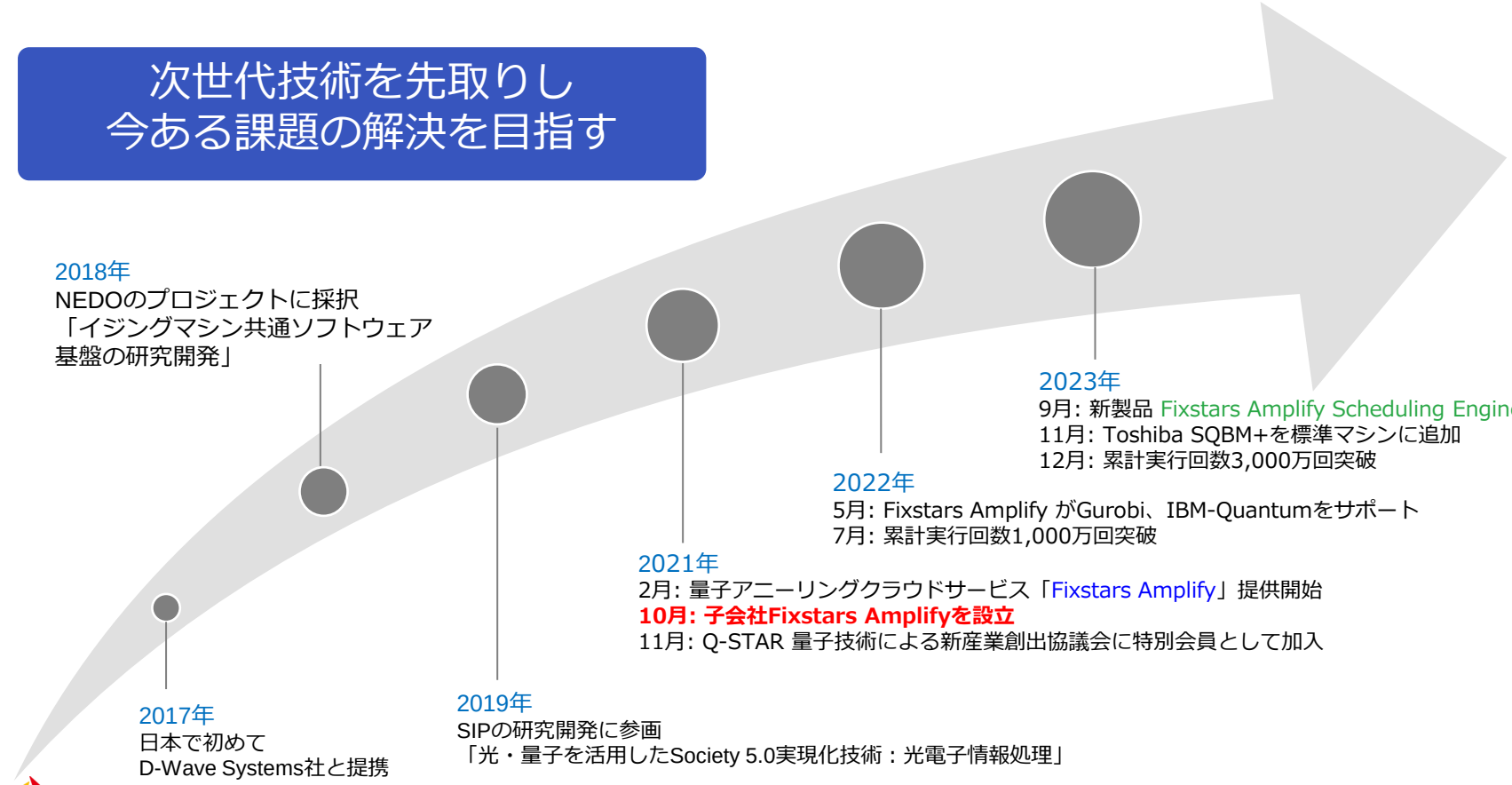
お客様専任のエンジニアが直接ヒアリングを行い、**高速化**を実現するために乗り越えるべき課題や問題を明確にしていきます。

高速化のワークフロー



フィックスターズの量子・最適化技術への取り組み

次世代技術を先取りし
今ある課題の解決を目指す



2017年
日本で初めて
D-Wave Systems社と提携

2018年
NEDOのプロジェクトに採択
「イジングマシン共通ソフトウェア基盤の研究開発」

2019年
SIPの研究開発に参画
「光・量子を活用したSociety 5.0実現化技術：光電子情報処理」

2021年
2月: 量子アニーリングクラウドサービス「Fixstars Amplify」提供開始
10月: 子会社Fixstars Amplifyを設立
11月: Q-STAR 量子技術による新産業創出協議会に特別会員として加入

2022年
5月: Fixstars Amplify がGurobi、IBM-Quantumをサポート
7月: 累計実行回数1,000万回突破

2023年
9月: 新製品 Fixstars Amplify Scheduling Engine リリース
11月: Toshiba SQBM+を標準マシンに追加
12月: 累計実行回数3,000万回突破

FIXSTARS Amplify

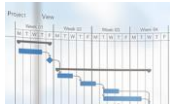
Fixstars Amplify: 製品ポートフォリオ

Fixstars Amplify
(2021/2～)

- **汎用的な問題**に対応する SDK と実行環境 (AE)
- 2次の QUBO 問題が得意
- シフト最適化、経路最適化、設計変数最適化向けに自分で**自由に定式化**した目的関数や制約条件の中で一番良い組み合わせを高速・高精度に探索
- 量子アニーリング・イジングマシンを活用してブラックボックス最適化問題を解くことも可能

Fixstars Amplify Scheduling Engine
(2023/9～)

- **スケジューリング問題に特化**した SDK と実行環境 (Scheduling Engine)
- 最適化の目的を Makespan の最小化に絞り、**定式化不要**で、複雑な制約条件を通常のプログラミングの延長で実装可能
- 生産計画・人員シフト計画・配送計画など幅広いスケジューリングに適用可能



本日のセミナーはこちら

様々な分野でFixstars Amplifyの利用が拡大しています

豊田通商株式会社 TOYOTA TSUSHO CORPORATION

NTT DATA

NRI 野村総合研究所 Nomura Research Institute

住友商事株式会社

登録社・組織数: 約650

累計実行回数: 3,500万回超

IPA Better Life with IT

Fusic

慶應義塾 Keio University

早稲田大学

株式会社ネクスティ エレクトロニクス

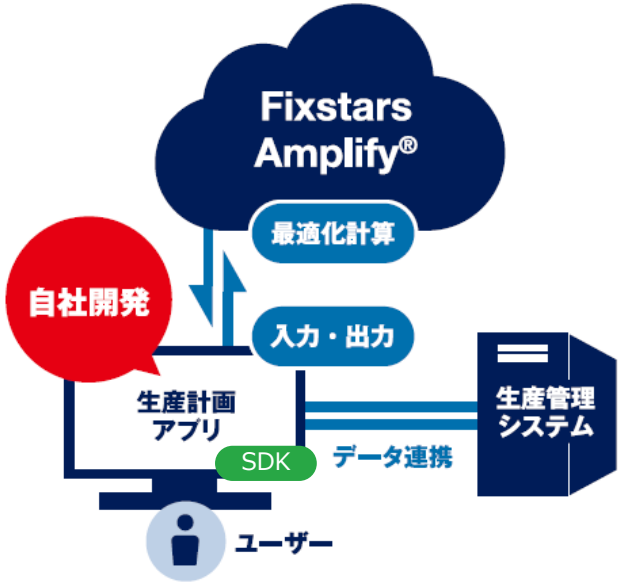
豊田中央研究所

Fixstars Amplify Scheduling Engine のご紹介

Fixstars Amplify Scheduling Engine とは

Fixstars Amplify Scheduling Engine (Amplify SE) は、プログラム開発を簡単に行えるソフトウェア開発支援キット (SDK) と、高精度な解を瞬時に求める高性能な計算エンジンによって、スケジュール最適化アプリをユーザー自ら構築できるクラウドサービスです

https://amplify.fixstars.com/scheduling

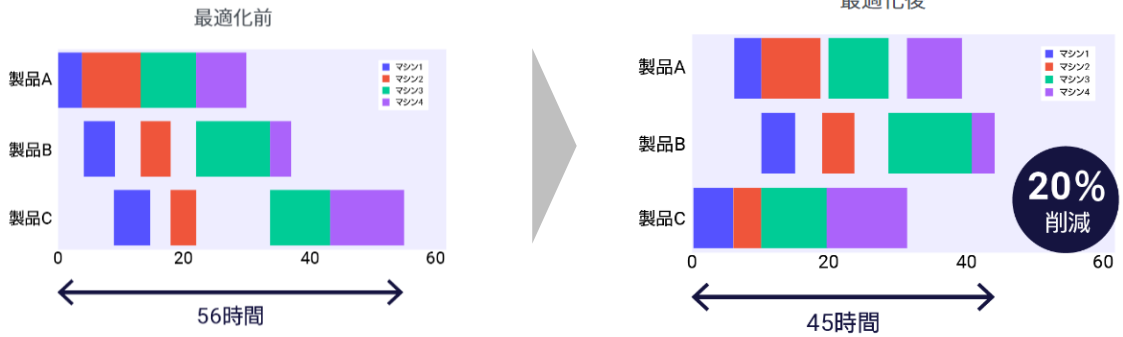


簡単	● SDKをインストールするだけですぐに使える (pip install amplify_sched) ● 目的関数や制約条件の定式化は不要で、通常のプログラムの延長で実装可能
高機能・高性能	● 複雑な制約条件でも簡単に最適化 ● コンピュータの性能を最大限に引き出すアルゴリズムによって、大規模なスケジューリング問題も瞬時に最適化
始めやすい	● 評価・検証用途には開発環境と実行環境が無償で利用可能 ● 多くのチュートリアル、サンプルコードを整備・拡充

Amplify SE によるスケジューリング問題の最適化

Amplify SE なら、自社固有の状況や条件を考慮して、生産性の最大化や在庫を最小化する計画を瞬時に立案できます

https://amplify.fixstars.com/ja/scheduling/product



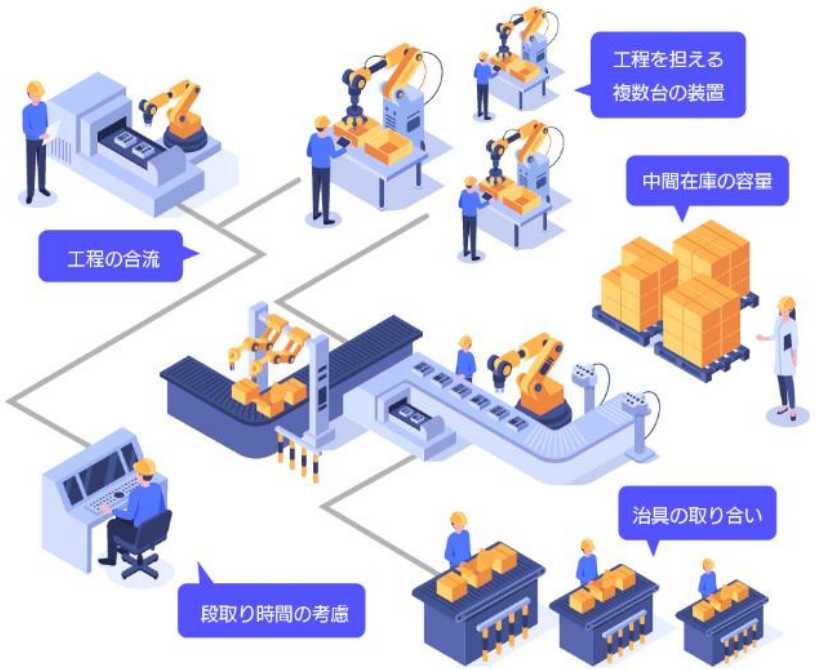
実装のイメージ

```python
from amplify_sched import *
model = Model()
model.jobs.add("Job A")
model.machines.add("Machine X")
model.jobs["Job A"].append(Task())
model.jobs["Job A"][0].processing_times["Machine X"] = 10
model.jobs["Job A"][0].release_time = 10
model.jobs["Job A"][0].deadline = 20
gantt = model.solve(token=token, timeout=1)
gantt.timeline()
```

マシンや様々な条件等の情報を model に追加していき、最後に model をマシンに投げて求解します (目的は Makespan (最終生産完了時間) の最小化)

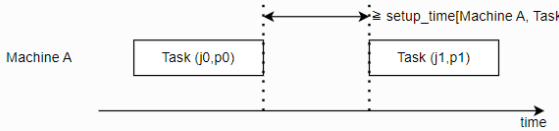
ソフトウェア開発支援キット

Amplify SE には、実際の現場にある工程の合流・分割、在庫置き場の容量、段取り時間など複雑な制約条件がサポートされており、簡単に実装できます

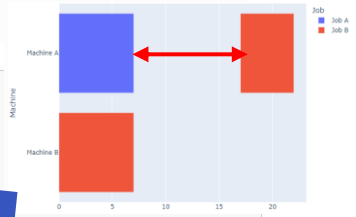


setup_time

ある Machine において、ある Task からある Task への切り替えにかかる準備時間(setup_time)を設定できます。準備時間が設定されている場合、その時間より短い間隔では次の Task を開始できません




```
model = Model()
model.jobs.add("Job A")
model.jobs.add("Job B")
model.machines.add("Machine A")
model.machines.add("Machine B")
model.jobs["Job A"].append(Task())
model.jobs["Job A"][0].processing_times["Machine A"] = 7
model.jobs["Job B"].append(Task())
model.jobs["Job B"][0].processing_times["Machine B"] = 7
model.jobs["Job B"].append(Task())
model.jobs["Job B"][1].processing_times["Machine A"] = 5
model.machines["Machine A"].setup_times.append((10, model.jobs["Job A", 0], model.jobs["Job B", 1]))
model.machines["Machine A"].setup_times.append((10, model.jobs["Job B", 1], model.jobs["Job A", 0]))
gantt = model.solve(token=token, timeout=1)
gantt.timeline(machine_view=True)
```

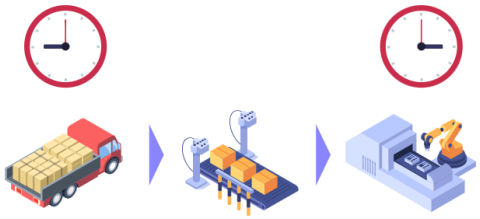


ソフトウェア開発支援キット

様々な条件を表現するため便利な[機能]が多数用意されています

開始可能時刻と終了締め切り時刻

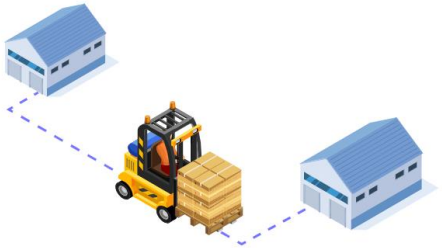
ドキュメントを見る



- 原料が届く時刻が決まっているため、届くまで仕事を始めることができない
- 納期が決まっているため、それまでに仕事を終わらせないといけない

マシン間の輸送時間

ドキュメントを見る



工程の間に、ある場所からある場所へ移動する時間が必要

マシンが稼働できない時間区間

ドキュメントを見る

定期メンテナンスのため、装置を使えない時間がある

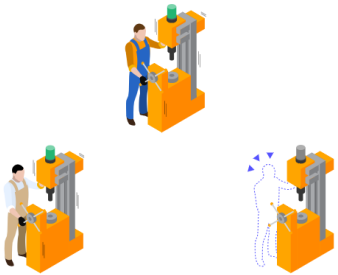
次の処理が始まるまで保管する倉庫

ドキュメントを見る

中間在庫置き場のスペースに限りがある

タスクの実行に必要なリソース

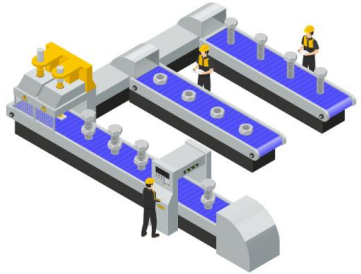
ドキュメントを見る



複数のタスクを、限られた人員が付きっきりで動かす必要がある

依存するジョブ

ドキュメントを見る



それぞれ別の工程で作られた部品を組み合わせて完成する

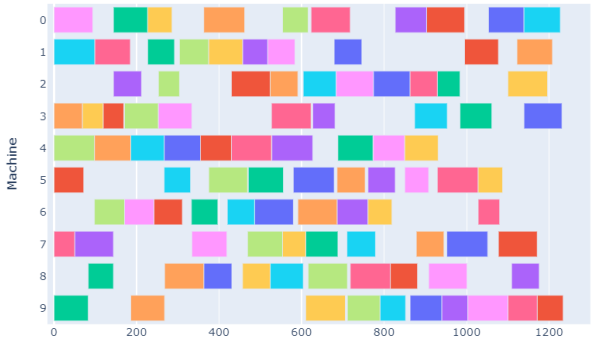
高性能な計算エンジン

クラウドにある最先端の高性能コンピュータ群と、コンピュータの性能を最大限に引き出すアルゴリズムによって、大規模なスケジューリング問題も瞬時に最適化計算ができます

https://amplify.fixstars.com/ja/scheduling/product#engine

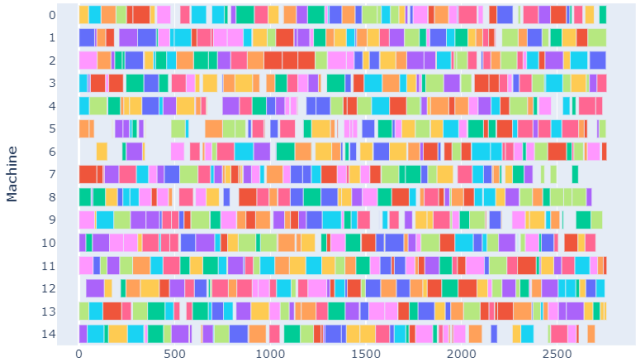
シンプルな問題

ベンチマーク
abz5
マシン数
10
タスク数
100
実行時間
1秒
最適解到達率
100% 最適解



複雑な問題

ベンチマーク
ta51
マシン数
15
タスク数
750
実行時間
2分
最適解到達率
100% 最適解



無料で始めることができて、学習コンテンツも豊富

無料で始められるクラウドサービスなので、思い立ったその日からご利用いただけます。ウェブサイトにチュートリアルやサンプルコードなど豊富な学習コンテンツをご用意していますので、初心者の方でも簡単に始められます

https://amplify.fixstars.com/ja/scheduling/resources

Amplify SE のはじめ方

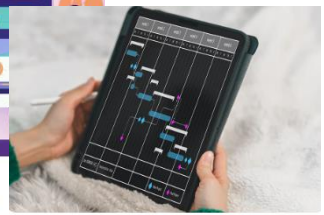
1. **ユーザー登録と無料トークンの入手**
   無料トークンはユーザー登録と同時に発行されます。
2. **Amplify SEを使うためのPython環境の準備**
   Amplify SEを利用するためのPython環境を準備します。
3. **Amplify SEを使ってプログラムを作って実行**
   1. **今すぐ動かしたい方**
   2. **Google Colaboratoryで試してみたい方** (Google Colaboratoryを使用します)
   3. **本格的な開発をしたい方** (Python環境 + Visual Studio Codeを使用します)

チュートリアル

1. SE入門チュートリアル

Amplify SEへようこそ。こちらチュートリアルを始めてください。始めにジューリング例を使って、Amplみについて説明します。

詳しく見る →



4. より柔軟な生産計画立案（実装例）

あるクッキー工場を題材とし、各タスクを処理可能なマシン候補が複数ある場合の生産計画立案について考えます。

詳しく見る →

サンプルコード

マシニングセンタへのジ

必要工具の異なる様々な材料加工具交換装置搭載の加工機械でジョブ割り当てについて考えます。

詳しく見る →

配送スケジューリング

Amplify SEの応用例として、荷物輸送や旅客送迎などの配送スケジューリングについて取り扱います。

詳しく見る →

生産計画最適化のワークショップ
～事前準備～

ワークショップの事前準備（１）

- ご自身の PC のブラウザ上で Python のプログラミングを行います。Google Colaboratory を使うので、事前に Google Colaboratory にログインできることをご確認ください（Google アカウントが必要です）。

Google Colab 検索

https://colab.research.google.com/

- Fixstars Amplify の無料トークンの取得有無をご確認ください。まだの方は、こちら からユーザー登録をして無料トークンを取得してください（1分で完了します）。

Fixstars Amplify 検索

https://amplify.fixstars.com/

ワークショップの事前準備（2）

取得された Amplify SE の無料トークンを用いてトークンチェック用のサンプルコードが動くか、以下のステップでご確認をお願いします。

1. 以下の URL にアクセスしてください。サンプルコードは閲覧のみ可能な状態なので、「ファイル」→「ドライブにコピーを保存」して、ご自身の Google ドライブにコピーを作成してください。
   https://colab.research.google.com/drive/1sFPSn7CGNX4i4BsGLxC4SDdVVtwSkLLG#scrollTo=5WBioiZf5A5y

2. コピーしたファイルの1番目のセルにご自身の無料トークンを入力してください（***印の部分を書き換えてください）。ご自身の無料トークンは、「アクセストークン」ページの「Fixstars Amplify SE」のセクションでご確認いただけます。トークンを入力後、再生ボタンまたは Shift +Enter で1番目のセルを実行して下さい。

```
token = ”**********************************”  # ご自身のトークンを入力
```

3. 1番目のセルの実行が完了したら、2番目のセルも再生ボタンまたは Shift + Enter で実行してください。ガントチャートが出力されれば OK です！

ワークショップの事前準備（3）

- ワークショップで使うサンプルコードを以下の URL より取得して下さい。サンプルコードは閲覧のみ可能な状態なので、「ファイル」→「ドライブにコピーを保存」して、ご自身の Google ドライブにコピーを作成してください。
- サンプルコードの1番目のセルにはご自身の Amplify SE のトークンを入力いただく必要があります。

➢ サンプルコード
https://colab.research.google.com/drive/1aKewDnJb3j0dyrpmpj0T22MssUprFo_5?hl=ja#scrollTo=_6qEc5kvko4C

質問は随時、チャット か Q&A でお願いします。
対応可能なメンバーが対応致します。

生産計画最適化のワークショップ

ワークショップ: 本日取り扱う問題

https://amplify.fixstars.com/ja/scheduling/resources#code

3

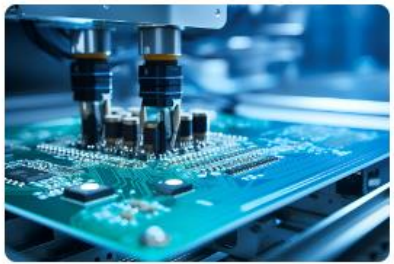
サンプルコード

ウェブブラウザ上で、様々なスケジューリング問題のサンプルコードを実行することができます。

マシニングセンタへのジョブ割り当て

必要工具の異なる様々な材料加工を、自動工具交換装置搭載の加工機械で処理する際のジョブ割り当てについて考えます。

詳しく見る →



プリント回路基盤の生産スケジューリング

複雑で様々な制約を伴う工程から構成されるプリント回路基板製造の最適スケジューリングについて考えます。

詳しく見る →

配送スケジューリング

Amplify SEの応用例として、荷物輸送や旅客送迎などの配送スケジューリングについて取り扱います。

詳しく見る →

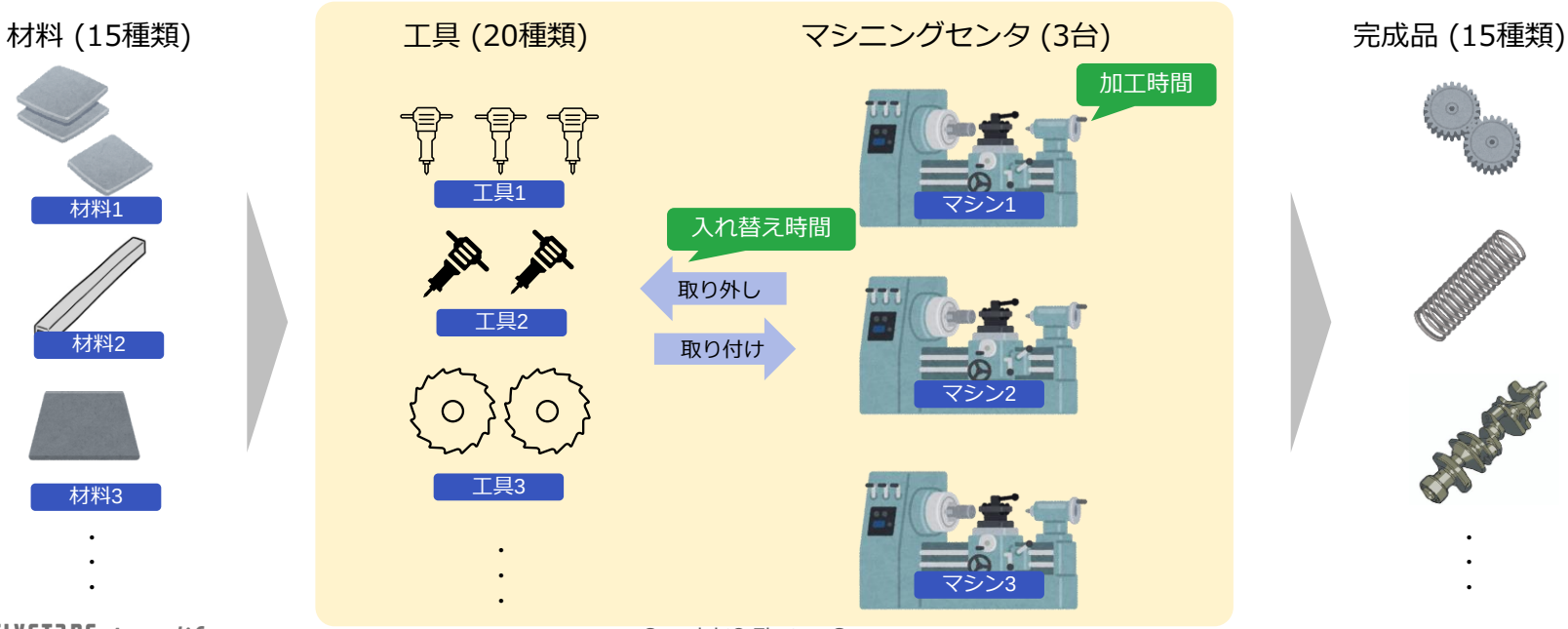
マシニングセンタとは

マシニングセンタ（machining center）は、回転工具を使用してフライス削りや中ぐり、穴あけ、ねじ立てといった切削加工を、1台で行える工作機械

(出展: https://www.dmgmori.co.jp/products/machines/)

ワークショップ: 問題設定

「3台のマシニングセンタを使って15種類の材料を加工して15個の完成品を作ります。マシニングセンタによって、同じ材料を加工するのにかかる「加工時間」は異なります。また、加工する材料に応じて計20種類の工具の中から必要な工具をマシニングセンタに取り付ける必要があり、工具の取り外し・取り付けのための「入れ替え時間」がかかります。各工具の在庫は2〜3個あり、同じ工具を必要とする材料もあります。工具をやりくりしながら、加工時間と入れ替え時間を考慮して、最終の製品の完了時間が一番短くなる効率的な計画を立てます



ワークショップ: 必要な入力情報

「3台のマシニングセンタを使って15種類の材料を加工して15個の完成品を作ります。マシニングセンタによって、同じ材料を加工するのにかかる「加工時間」は異なります。また、加工する材料に応じて計20種類の工具の中から必要な工具をマシニングセンタに取り付ける必要があり、工具の取り外し・取り付けのための「入れ替え時間」がかかります。各工具の在庫は2～3個あり、同じ工具を必要とする材料もあります。工具をやりくりしながら、加工時間と入れ替え時間を考慮して、最終の製品の完了時間が一番短くなる効率的な計画を立てます

使用工具

	工具 1	工具 2	工具 3	工具 4	・・・
材料 1	o	o	o	o	
材料 2					
材料 3					
材料 4					
材料 5	o	o			
材料 6			o	o	

・
・
・

入れ替え時間 =
工具の取り外し数 × 取り外し時間 +
工具の取り付け数 × 取り付け時間

※ 工具当たりの取り外し・取り付け時間は10とします

工具の数 (同時使用可能数)

工具	工具 1	工具 2	工具 3	工具 4	工具 5	・・・
同時使用可能数	2	2	3	3	3	

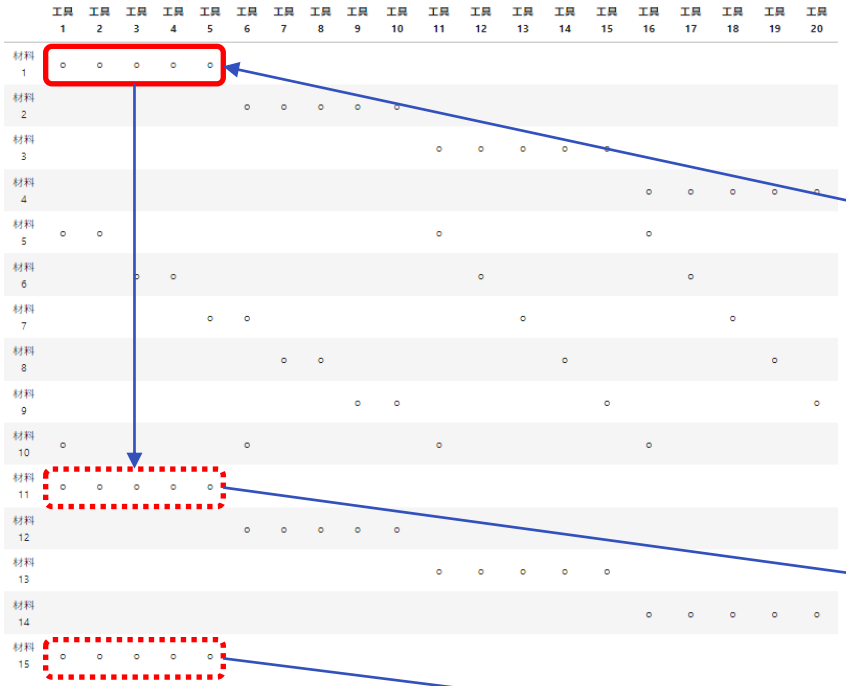
加工時間

	マシン 1	マシン 2	マシン 3
材料 1	10	20	30
材料 2	20	30	10
材料 3	30	40	20
材料 4	40	10	30
材料 5	10	30	40
材料 6	20	40	10
材料 7	30	10	20
材料 8	40	20	30
材料 9	10	20	40
材料 10	20	30	10
材料 11	30	40	20
材料 12	40	10	30
材料 13	10	30	40
材料 14	20	40	10
材料 15	30	10	20

最適化は機械学習とは異なり、大量が学習データは不要です

ワークショップ: 人の手でやってみよう

使用工具



工具の数 (同時使用可能数)

工具	工具 1	工具 2	工具 3	工具 4	工具 5	工具 6	工具 7	工具 8	工具 9	工具 10	工具 11	工具 12	工具 13	工具 14	工具 15	工具 16	工具 17	工具 18	工具 19	工具 20
同時使用可能数	2	2	3	3	3	2	2	2	2	2	2	2	2	2	2	2	3	2	2	2

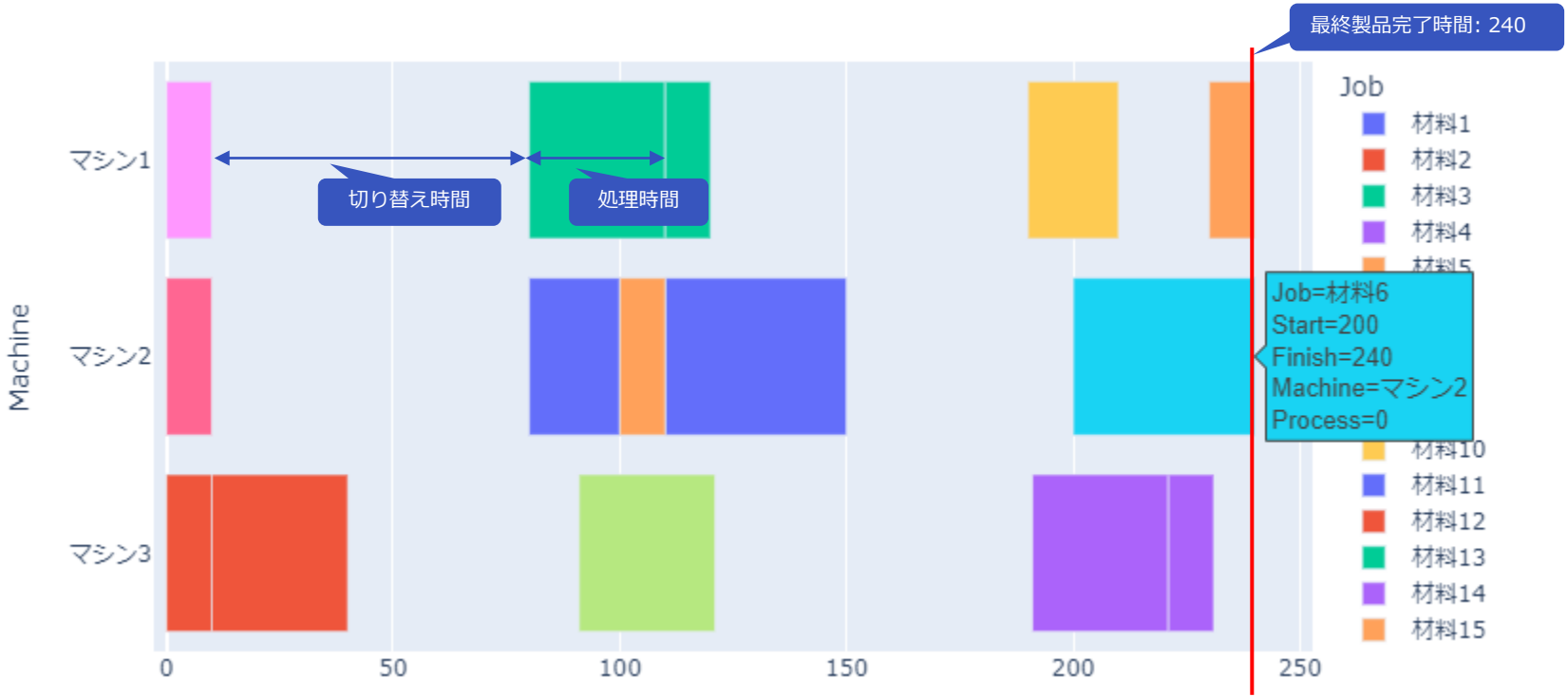
加工時間

	マシン 1	マシン 2	マシン 3
材料 1	10	20	30
材料 2	20	30	10
材料 3	30	40	20
材料 4	40	10	30
材料 5	10	30	40
材料 6	20	40	10
材料 7	30	10	20
材料 8	40	20	30
材料 9	10	20	40
材料 10	20	30	10
材料 11	30	40	20
材料 12	40	10	30
材料 13	10	30	40
材料 14	20	40	10
材料 15	30	10	20

まずは、「材料1」を処理時間が短い「マシン1」に割り当ててみよう、、「工具1～5」を「マシン1」に割り当てて、、「材料11」と「材料15」は同じ「工具1～5」を使うから、同じ「マシン1」に割り当てれば切り替え時間が必要ないから連続で作った方がいいな、、いや、、それとも、切り替え時間がかかったとしても、処理時間が短い別のマシンでやった方がいいのか、、、う～ん、それぞれの場合でその先もやってみるしかないか、、、組合せ多すぎじゃないか、、これを人の手でやるの無理じゃないか、、、

FIXSTARS Amplify Scheduling Engine

ワークショップ: 一つの答え



ワークショップ: 解くための 3 steps

「3台のマシニングセンタを使って15種類の材料を加工して15個の完成品を作ります。マシニングセンタによって、同じ材料を加工するのにかかる「加工時間」は異なります。また、加工する材料に応じて計20種類の工具の中から必要な工具をマシニングセンタに取り付ける必要があり、工具の取り外し・取り付けのための「入れ替え時間」がかかります。各工具の在庫は2〜3個あり、同じ工具を必要とする材料もあります。工具をやりくりしながら、加工時間と入れ替え時間を考慮して、最終の製品の完了時間が一番短くなる効率的な計画を立てます

全てを一度に作るのは難しいので、3つの Step に分けてプログラムを作ります

Step1

入力情報の準備
- 各材料の各マシンでの加工時間のデータフレームを作成
- 各材料の加工に必要な工具の情報をデータフレームに追加
- 各工具の個数（同時使用可能数）のデータフレームを作成

Step2

resource を使います (ドキュメントはこちら)

モデルの構築
- モデルを作成しマシン、ジョブ、リソースの情報を追加
- 切り替え時間を計算して各マシンに追加

setup_time を使います (ドキュメントはこちら)

Step3

マシンで求解して可視化
- SE に問題を投げて求解し、得られた解を可視化

Step1

入力情報の準備

- 各材料の各マシンでの加工時間のデータフレームを作成
- 各材料の加工に必要な工具の情報をデータフレームに追加
- 各工具の個数（同時使用可能数）のデータフレームを作成

各材料の各マシンでの加工時間のデータフレーム

材料	加工時間(マシン1)	加工時間(マシン2)	加工時間(マシン3)
材料1	10	20	30
材料2	20	30	10
材料3	30	40	20
材料4	40	10	30
材料5	10	30	40
材料6	20	40	10
材料7	30	10	20
材料8	40	20	30
材料9	10	20	40
材料10	20	30	10
材料11	30	40	20
材料12	40	10	30
材料13	10	30	40
材料14	20	40	10
材料15	30	10	20

実装

```python
import pandas as pd  # Python のデータ構造やデータ処理を行うデータ解析ライブラリ

machine_list = ["マシン1", "マシン2", "マシン3"]  # マシニングセンタのリスト

# (1) 材料の情報をまとめたDataFrameを構築する
df_material = pd.DataFrame()

# 材料の名前
df_material["材料"] = ["材料{}".format(i) for i in range(1, 16)]
df_material.set_index("材料", inplace=True)

# 各マシンによる各材料の加工時間
df_material["加工時間(マシン1)"] = [
    10,
    20,
    30,
    40,
    10,
    20,
    30,
    40,
    10,
    20,
    30,
    40,
    10,
    20,
    30,
]
```

・
・
・

```python
remove_work_time = 10  # 工具の取り外し時間
set_work_time = 10  # 工具の取り付け時間
```

今回のワークショップでは、取り外しも取り付けも、
工具あたりにかかる時間を 10 とします

Step1

入力情報の準備

- 各材料の各マシンでの加工時間のデータフレームを作成
- 各材料の加工に必要な工具の情報をデータフレームに追加
- 各工具の個数（同時使用可能数）のデータフレームを作成

	工具1	工具2	工具3	工具4	工具5	工具6	工具7	工具8	工具9	工具10	工具11	工具12	工具13	工具14	工具15	工具16	工具17	工具18	工具19	工具20
材料1	o	o	o	o	o															
材料2						o	o	o	o	o										
材料3											o	o	o	o	o					
材料4																o	o	o	o	o
材料5	o	o									o					o				
材料6			o	o								o					o			
材料7					o	o							o					o		
材料8							o	o						o					o	
材料9									o	o					o					o
材料10	o					o					o					o				
材料11	o	o	o	o	o															
材料12						o	o	o	o	o										
材料13											o	o	o	o	o					
材料14																o	o	o	o	o
材料15	o	o	o	o	o															

実装

```
# 材料1,...,15に使用する工具
df_material["使用工具"] = [
    ["工具1", "工具2", "工具3", "工具4", "工具5"],  # 材料1
    ["工具6", "工具7", "工具8", "工具9", "工具10"],  # 材料2
    ["工具11", "工具12", "工具13", "工具14", "工具15"],  # 材料3
    ["工具16", "工具17", "工具18", "工具19", "工具20"],  # 材料4
    ["工具1", "工具2", "工具11", "工具16"],  # 材料5
    ["工具3", "工具4", "工具12", "工具17"],  # 材料6
    ["工具5", "工具6", "工具13", "工具18"],  # 材料7
    ["工具7", "工具8", "工具14", "工具19"],  # 材料8
    ["工具9", "工具10", "工具15", "工具20"],  # 材料9
    ["工具1", "工具6", "工具11", "工具16"],  # 材料10
    ["工具1", "工具2", "工具3", "工具4", "工具5"],  # 材料11
    ["工具6", "工具7", "工具8", "工具9", "工具10"],  # 材料12
    ["工具11", "工具12", "工具13", "工具14", "工具15"],  # 材料13
    ["工具16", "工具17", "工具18", "工具19", "工具20"],  # 材料14
    ["工具1", "工具2", "工具3", "工具4", "工具5"],  # 材料15
]

df_material
```

Step1

入力情報の準備

- 各材料の各マシンでの加工時間のデータフレームを作成
- 各材料の加工に必要な工具の情報をデータフレームに追加
- 各工具の個数（同時使用可能数）のデータフレームを作成

工具	工具1	工具2	工具3	工具4	工具5	工具6	工具7	工具8	工具9	工具10	工具11	工具12	工具13	工具14	工具15	工具16	工具17	工具18	工具19	工具20
同時使用可能数	2	2	3	3	3	2	2	2	2	2	2	2	2	2	2	2	3	2	2	2

実装

```
# (2) 工具の情報をまとめたDataFrameを構築する
df_tool = pd.DataFrame()

# 工具の名前
df_tool["工具"] = ["工具{}".format(i) for i in range(1, 21)]
df_tool.set_index("工具", inplace=True)

# 同時使用可能数
df_tool["同時使用可能数"] = [
    2,
    2,
    3,
    3,
    3,
    2,
    2,
    2,
    2,
    2,
    2,
    2,
    2,
    2,
    2,
    3,
    2,
    2,
    2,
] # 工具1, 2, ..., 20

df_tool
```

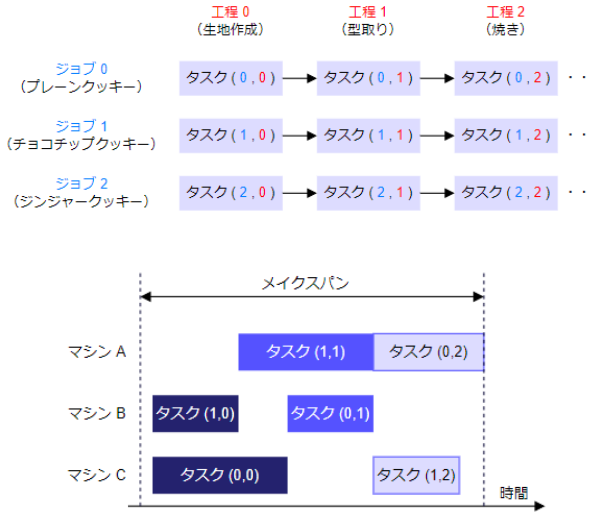
Step2

モデルの構築

- モデルを作成しマシン、リソース、ジョブの情報を追加
- 切り替え時間を計算して各マシンに追加

まず、モデルを構築するにあたり必要な用語の整理をします

https://amplify.fixstars.com/ja/scheduling/resources/tutorial/se_concept



今回のワークショップでは以下に対応します

※ 今回のワークショップでは、マシンを移動することなく、マシニングセンタ内ですべての工程を行うことを想定しているため、全てのジョブでタスクは1つになります

要素	Amplify SE の言葉	
マシニングセンタ	マシン	machine1 (マシン1) - machine3 (マシン3)
材料	ジョブ	job1 (材料1) - job15 (材料15)
工具	リソース	resource1 (工具1) – resource20 (工具20)

Step2

モデルの構築

- モデルを作成しマシン、リソース、ジョブの情報を追加
- 切り替え時間を計算して各マシンに追加

実装

```
from amplify_sched import * # インストールした Amplify SE からすべてのライブラリをインポート
import itertools # 効率的なループを実現するためのイテレータ関数を提供する Python 標準のライブラリ

# (1) まずモデルを定義。このモデルに様々な情報を追加していく
model = Model()

# (2) モデルにマシン（マシン1、マシン2、等）を追加
for m in machine_list:
    model.machines.add(m)
print(model.machines)
```

```
# (3) モデルにリソース（工具1、工具2、等）と同時使用可能数を追加
for k in df_tool.index:
    model.resources.add(k)
    model.resources[k].capacity = int(df_tool.loc[k, "同時使用可能数"])
print(model.resources)
```

```
# (4) モデルにジョブ（材料1、材料2、等）を追加
for j in df_material.index:
    model.jobs.add(j)
    model.jobs[j].append(Task())

    # モデルに各ジョブの各マシンでの加工時間を追加
    for m in machine_list:
        model.jobs[j][0].processing_times[m] = int(df_material.loc[j, f"加工時間({m})"])

    # モデルに各ジョブに必要なリソースを追加
    for k in df_material.loc[j, "使用工具"]:
        model.jobs[j][0].required_resources.append(k)
print(model.jobs)
```

各材料の各マシンでの加工時間を processing_times に格納

各材料の加工に必要な工具を required_resources に追加

Step2

モデルの構築

- モデルを作成しマシン、リソース、ジョブの情報を追加
- 切り替え時間を計算して各マシンに追加

	工具1	工具2	工具3	工具4	工具5	工具6	工具7	工具8	工具9	工具10	工具11	工具12	工具13	工具14	工具15	工具16	工具17	工具18	工具19	工具20
材料1	o	o	o	o	o															
材料2						o	o	o	o	o										
材料3											o	o	o	o	o					
材料4																o	o	o	o	o
材料5	o	o									o					o				

材料1 → 材料5: (取り外し3 + 取りつけ2) * 10 = 50
材料15 → 材料1: (取り外し2 + 取り付け3) * 10 = 50

実装

```
# (5) リソースの切り替え時間を計算して各マシンの setup_times に追加
for j1, j2 in itertools.combinations(df_material.index, 2):
    remove_num = len(
        set(df_material.loc[j1, "使用工具"]) - set(df_material.loc[j2, "使用工具"])
    )
    add_num = len(
        set(df_material.loc[j2, "使用工具"]) - set(df_material.loc[j1, "使用工具"])
    )
    time_j1_to_j2 = remove_num * remove_work_time + add_num * set_work_time
    time_j2_to_j1 = add_num * remove_work_time + remove_num * set_work_time

    # 切り替え時間を setup_times に追加
    for m in machine_list:
        model.machines[m].setup_times.append((time_j1_to_j2, j1, j2))
        model.machines[m].setup_times.append((time_j2_to_j1, j2, j1))

print(model.machines["マシン1"].setup_times)
print(model)
```

材料_aから材料_bに入れ替える際に取り外す工具の数をカウント

材料_aから材料_bに入れ替える際に取り付ける工具の数をカウント

入れ替え時間を計算
(remove_work_time と set_work_time はいずれも 10)

材料_a → 材料_b と 材料_b → 材料_b の工具の入れ替えにかかる時間を各マシンの setup_times に追加

Step3

マシンで求解

SE で求解して得られた解を可視化

実装

```
###################################
# Step3: Amplify SE による求解と可視化
###################################

result = model.solve(token=token, timeout=10)  # タイムアウトは 10 秒
```

【ご参考】解の情報

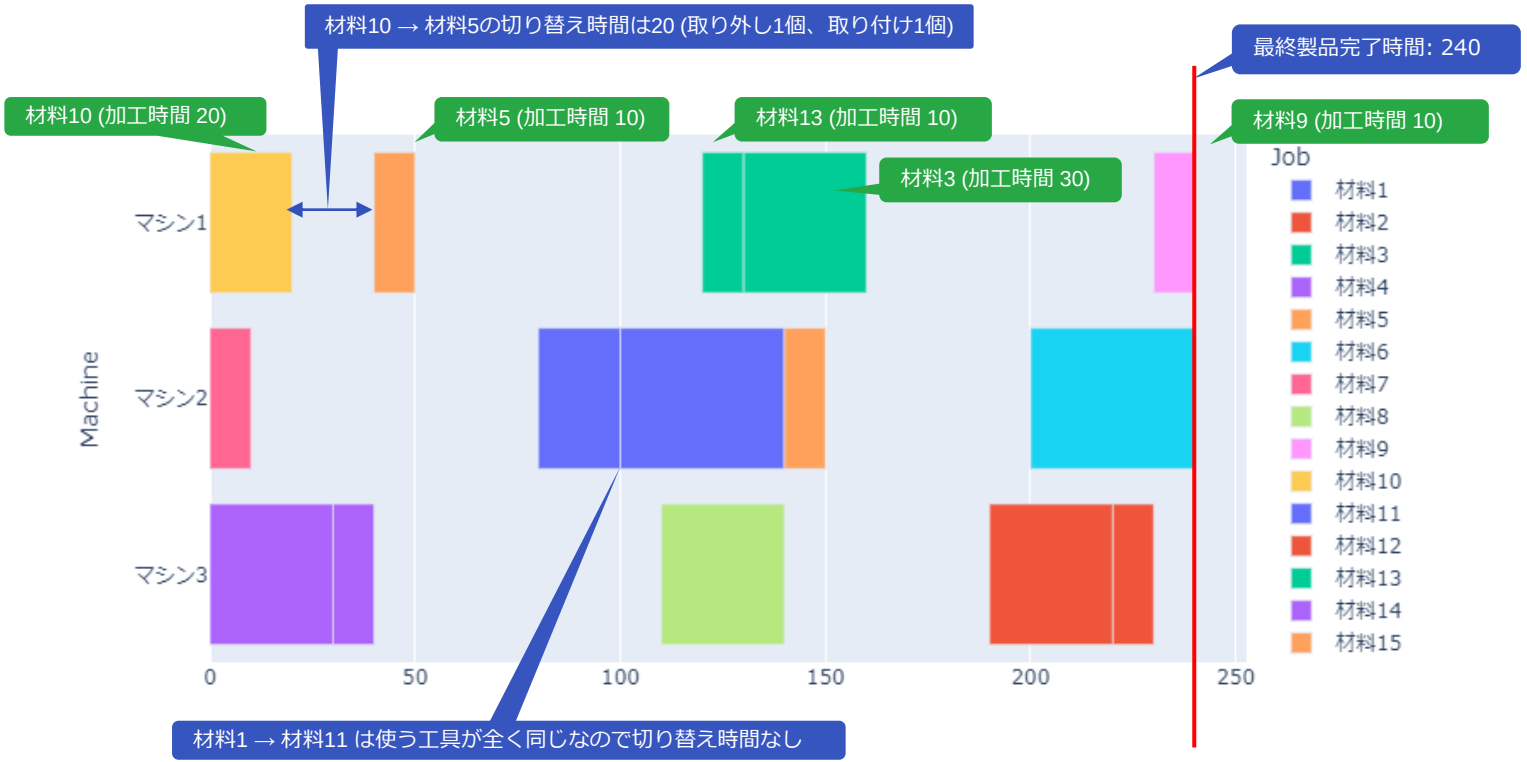
```
print(result.status)
```

status	意味
OPTIMAL	最適解
FEASIBLE	実行可能解（制約は満たすがメイクスパンは最小とは限らない）
INFEASIBLE	実行可能解が見つからなかった

Step3

マシンで求解

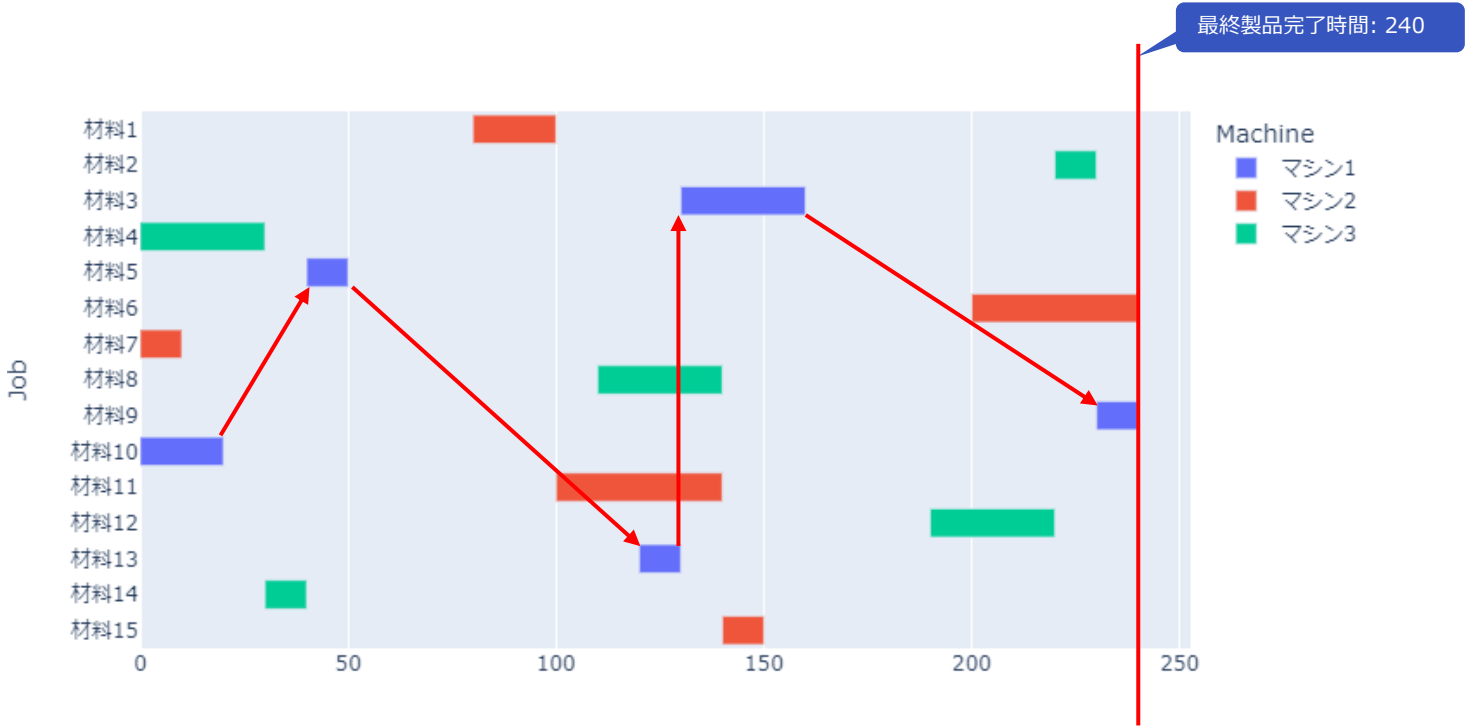
SE で求解して得られた解を可視化



Step3

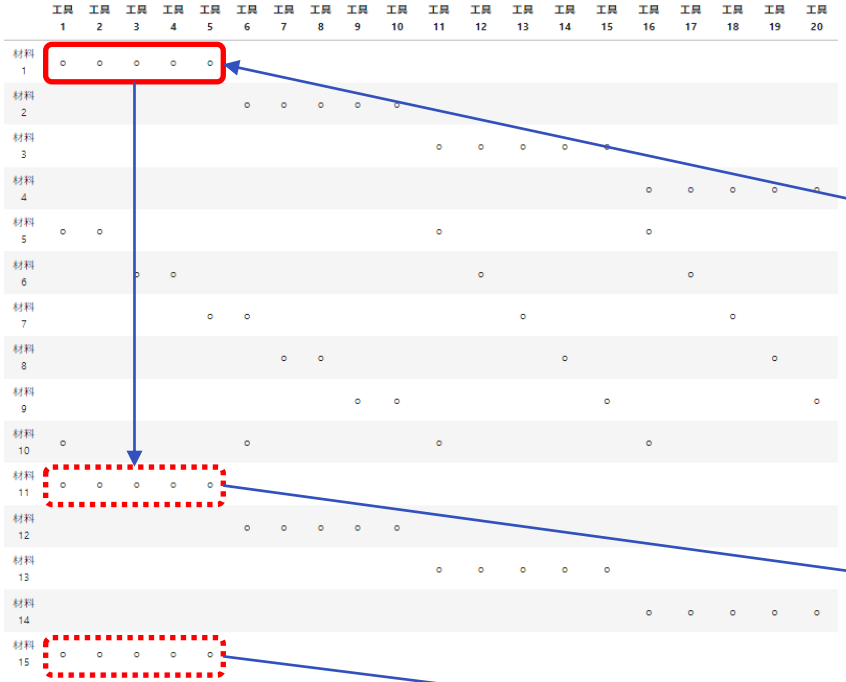
マシンで求解

SE で求解して得られた解を可視化



ワークショップ: おさらい

使用工具



工具の数 (同時使用可能数)

工具	工具 1	工具 2	工具 3	工具 4	工具 5	工具 6	工具 7	工具 8	工具 9	工具 10	工具 11	工具 12	工具 13	工具 14	工具 15	工具 16	工具 17	工具 18	工具 19	工具 20
同時使用可能数	2	2	3	3	3	2	2	2	2	2	2	2	2	2	2	2	3	2	2	2

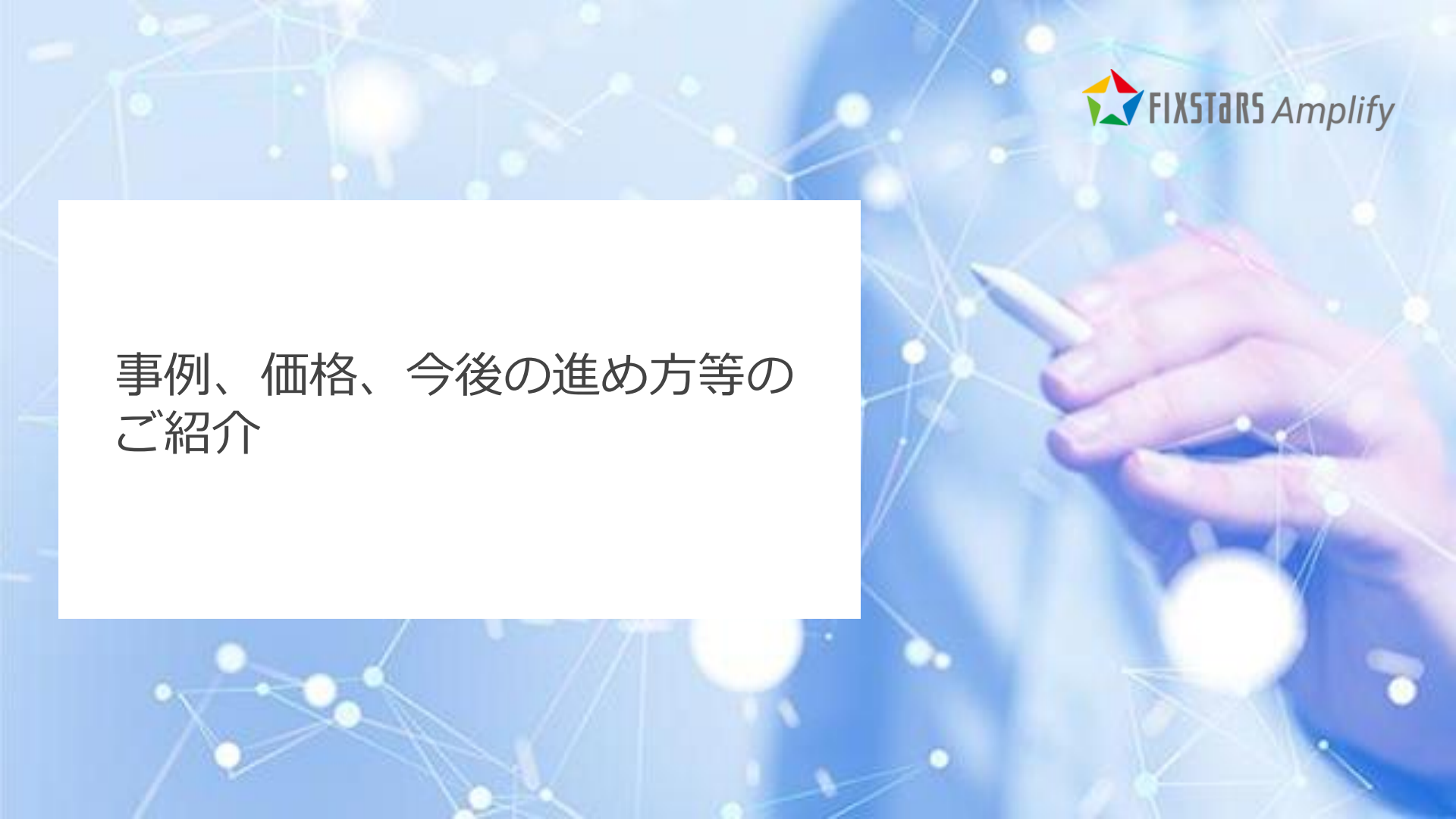
加工時間

	マシン 1	マシン 2	マシン 3
材料 1	10	20	30
材料 2	20	30	10
材料 3	30	40	20
材料 4	40	10	30
材料 5	10	30	40
材料 6	20	40	10
材料 7	30	10	20
材料 8	40	20	30
材料 9	10	20	40
材料 10	20	30	10
材料 11	30	40	20
材料 12	40	10	30
材料 13	10	30	40
材料 14	20	40	10
材料 15	30	10	20

まずは、「材料1」を処理時間が短い「マシン1」に割り当ててみよう、、「工具1〜5」を「マシン1」に割り当てて、、「材料11」と「材料15」は同じ「工具1〜5」を使うから、同じ「マシン1」に割り当てれば切り替え時間が必要ないから連続で作った方がいいな、、いや、、それとも、切り替え時間がかかったとしても、処理時間が短い別のマシンでやった方がいいのか、、、う〜ん、それぞれの場合でその先もやってみるしかないか、、、組合せ多すぎじゃないか、、これを人の手でやるの無理じゃないか、、、

FIXSTARS Amplify Scheduling Engine

事例、価格、今後の進め方等のご紹介



組合せ最適化問題

膨大な選択肢から、**制約条件**を満たす、**ベスト**な解を探索する（組合せ最適化問題）

参考: 慶應義塾大学 田中宗 准教授 「量子コンピュータ最前線とイジングマシンの可能性」

組合せ最適化の取り組み事例

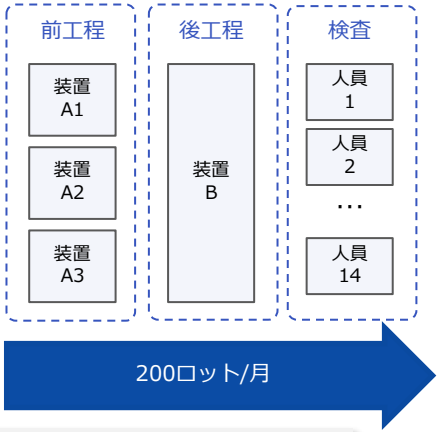
	業務	課題	成果
製造	生産計画	小ロット生産による計画の短サイクル化	計画自動化による短サイクルでの計画立案と最適化(リンク)
	生産計画	業務負荷大、属人化	生産計画の属人化解消と設備投資計画への活用(リンク)
	生産計画	段取回数・時間の最小化	段取回数を10%削減・属人化解消
	生産計画	製造人員の最小化	製造人員および人数変動の最小化
	人員シフト計画	スキルマッチ＋労働負荷平準化	従業員が納得する最適な人員シフトを実現(リンク)
	自動倉庫制御	トラックの荷積み待ちの渋滞	待ち時間の削減、Cyber Physical Systemの実現
物流	配車計画	労働時間短縮・属人化	長時間残業の解消
	荷積み計画	安全で効率的な荷積み	安全で効率的な荷積み計画
	倉庫作業シフト計画	スキルマッチ＋	作業効率の高いシフト、計画担当者のストレス軽減（リンク）
サービス業	人員シフト	属人化・業務負荷	従業員の希望を反映したシフト計画

生産計画の最適化により人員を14名から10名へ適正化

https://amplify.fixstars.com/ja/customers/interview/avaldata

生産状況
- 月間200ロットを生産
- 前工程⇒後工程⇒検査工程の3工程
- 物流の混乱による調達遅れ

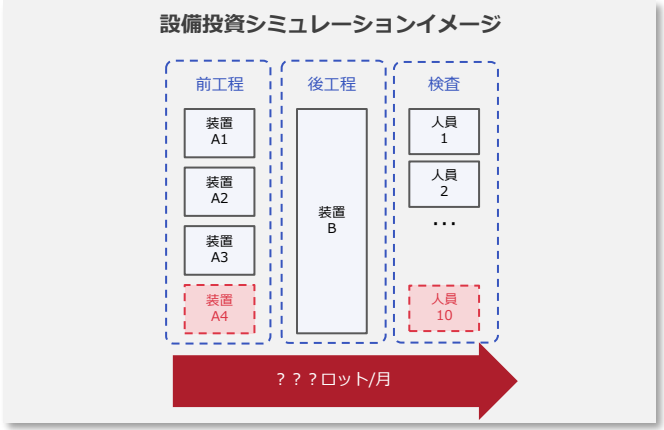
これまでの生産計画
- 計画の見直しに一日8時間（4時間/回×2回）掛かっていた
- 前工程、後工程の装置稼働率を上げる計画を考えていた
- 工場稼働日は休みを取れず、疲れが溜まっていた



FIXSTARS Amplify

成果
- 生産計画作成時間の短縮（8時間⇒20分）
- 属人化の解消
- 検査人員の適正化（ 14名⇒10名）

今後の発展
- 設備投資した際の生産量算出のシミュレーションに活用
- 営業担当による客先での納期回答
- 生産計画に合わせた原料納入日の自動調整



生産計画最適化 (電気機器製造メーカー A社様)

複数の製品事業部から様々なプリント基板の注文を受け、生産を行う部門

課題

生産する基板に応じて製造装置の部品や材料を交換する「段取り時間」が必要。段取り時間を考慮した効率的な生産スケジュールを作成したい
従来は、専任者が、一日数回・毎回数十分かけて経験に基づいてスケジュールを作成。更なる生産性向上やノウハウ継承のため、生産スケジュール作成の自動化に着手

FIXSTARS Amplify

効果

生産スケジュール作成の時間・コストの大幅な削減！
(一日あたり数時間 → 数分)

段取りのための製造装置の停止回数の削減！
(10%以上削減)

最適化未経験のご担当者様1人がプログラム試作開始から約1～2ヵ月間取り組んでこの効果を実現
現在は試作段階で、実運用に向けてモデルを改良中！

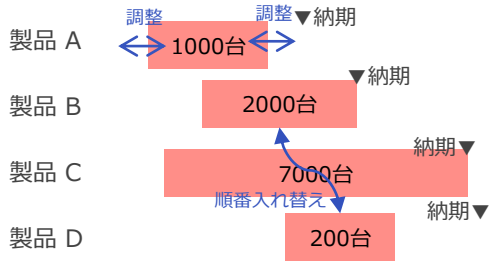
次期フェーズでは、Amplifyの活用領域の拡大を検討中！

生産計画最適化 (機器製造メーカー B社様)

各製品の生産数に応じて製造人員数の計画をする部門

課題

製品ごとに生産が開始できる日と納期が決まっていて、時期により日産数に差がある。製造に必要な人員数が緩やかに増減し、全体として最小化するように生産計画を立てている。専任者が2名で日々計画を立案・修正しているが、効率的な生産計画の立案、業務負荷削減、属人化解消のため自動化に着手。

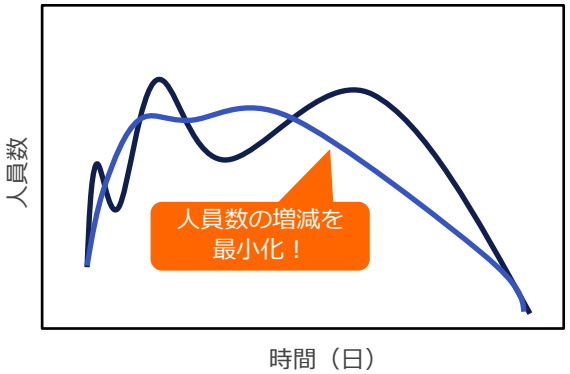


FIXSTARS Amplify

効果

必要人員数と日毎の人員数の増減の最小化！

計画立案の業務負荷削減・属人化解消！

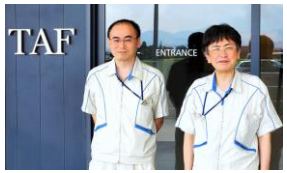


ユーザー事例

多くのお客様で成果が出始めています！

株式会社タアフ様

多品種少量生産の工程にマッチした生産計画アプリを開発



成果

- 自社の生産管理システムと連携した生産計画アプリを開発
- 設計初期から現場担当者と丁寧にすり合わせを行うことで、担当者が使いやすいアプリを実現
- 生産計画の標準作業化や生産リードタイムの短縮を目指す生産性向上プロジェクトにおいて、Amplify SEが最後のピースとなり生産計画自動化の道筋を得た

https://amplify.fixstars.com/ja/customers/interview/taf

株式会社アバールデータ様

生産計画の属人化解消と設備投資計画への活用を目指して

成果

- 生産計画を自動で立案・修正する機能を実現し、現場導入の道筋を得た
- 調達計画との連携や設備投資のシミュレーションによる経営判断への活用など、生産効率最適化のアイディアを得た

＊今後、現場導入に向けてプロジェクトを推進されています。

https://amplify.fixstars.com/ja/customers/interview/avaldata

様々な使い方が可能

Amplify SE で作った最適化プログラムは様々な使い方ができます！

01

既存システムと連携

既存のシステムとシームレスなデータ連携ができます。例えば、生産管理システムから必要なデータを取り出してAmplify SEで最適化計算をすることで、大規模な投資やシステム更新をせずに、高速かつ精密な生産計画の立案が実現できます。

02

エクセルから実行

最適化計算の結果をエクセルなどの表計算ソフトに出力できます。計画担当者の業務フローはこれまでと変える必要がありません。低コストで簡単に最適化のプロセスを管理・実行できるようになります。

03

独自アプリを開発

もちろん、新しいアプリをゼロから開発するという選択肢もあります。Amplify SEには、初心者でも理解しやすいチュートリアルや多くのサンプルコードが用意されています。ユーザーが自社で開発することもできますし、開発サポートも提供しています。

幅広い業務・業界に適用

急な変更に対応する

- 日次の生産計画（製造業）
- 日次の人員シフト計画（製造業・物流業）

効率的な計画を立てる

- 生産計画（製造業）
- 配送計画（物流業）
- 人員シフト計画（製造業）

資源を公平に配分する

- 人員シフト（製造業、物流業、小売業）
- プロフェッショナル人材サービス（製造業、介護）
- スパコンのジョブスケジューリング（IT）

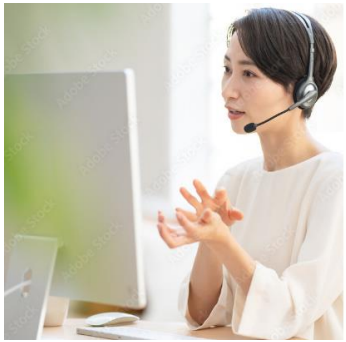
シミュレーションに役立てる

- 工事計画（建設業）
- 設備投資シミュレーション（製造業）

Amplify SE: クラウド利用料

https://amplify.fixstars.com/ja/scheduling/pricing

	ベーシックプラン まずは無料でお試しを	スタンダードプラン 実運用で使いたい方は	ビジネスプラン 組織単位のプラン
ご契約単位	個人	個人	組織
利用料金 (税抜)	**無料**	**月額10万円**	**月額20万円**
ご利用範囲	契約ユーザー	契約ユーザー	開発したアプリの利用者数に制限なし
計算環境	スモール	ラージ	ラージ (月間の累計実行時間を設定予定)
サポート	ベーシック	スタンダード	スタンダード



Plusオプション

料金：月額50万円（税込55万円）/ユーザー

- 問い合わせ回数は無制限
- ご質問には翌営業日までに回答（目安）
- 定式化・実装等のご相談
- 特別対応窓口や定例会の設置
- 特別技術支援※

※特別技術支援の内容に応じて期間等は個別にご相談

エンタープライズプラン

複数の組織や建物をカバーするライセンスや、長時間実行、オンプレ対応、クラウド上の専用環境の構築等の特別なご要望がある場合には個別に対応を検討させていただきます。お問い合わせよりお気軽にご相談ください。

- 開発・SI費用：個別見積もり
- 運用費用：月間100万円～ （目安）

詳しく見る

企業向けプライベートトレーニングのご紹介

https://amplify.fixstars.com/ja/seminar/private-training

企業向けプライベートトレーニング

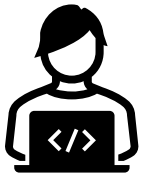
お客様が抱える実際の課題やデータを使った
カスタムメイドのトレーニングです！
(24万円/人(税抜))

全4回のレクチャーとお客様に実施いただく「課題」を含む約1.5か月のコースです。コースの前半では、量子アニーリング・イジングマシン専用の開発／実行環境であるFixstars Amplifyを用いてPython言語による組合せ最適化アプリケーション開発方法を学びます。後半では、お客様が抱える実際の課題やデータを使ったトレーニングを実施します。量子アニーリング・イジングマシンを使って実課題の解決に取り組んでみたい方に最適なコースです。

サンプルコードを使った座学

お客様の持ち込み課題を一緒に解きます

今後の進め方



自分で Amplify SE でプログラムを作ってみたい方

- ベーシックプラン（無料版）でまずはお試しを。チュートリアル、ドキュメントもご活用ください
- 有償プランでより大きな規模で本格開発（お困りごとがある場合は＋Plusオプションも追加可）
- 何かご質問やご不明点があれば、いつでもお問い合わせください！

PoC

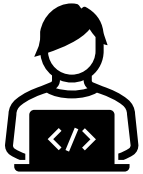
手を動かす前に自分の問題が解けるかざっと知りたい方

- 弊社または弊社が紹介する SIer が、無償 PoC や有償プロトタイプ開発を受託することも可能です。ご相談ください

Amplify SE を使ったシステムやアプリを開発してほしい方

- 弊社または弊社が紹介する SIer が、システム開発やアプリ開発を受託することも可能です。ご相談ください

今後の進め方



自分で Amplify SE でプログラムを作ってみたい方

ぜひこちらの問題も試してみてください！

プリント回路基盤の生産スケジューリング

複雑で様々な制約を伴う工程から構成されるプリント回路基板製造の最適スケジューリングについて考えます。

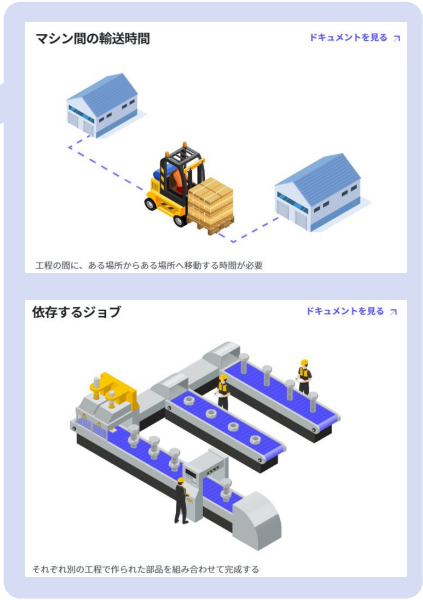
詳しく見る →

配送スケジューリング

Amplify SEの応用例として、荷物輸送や旅客送迎などの配送スケジューリングについて取り扱います。

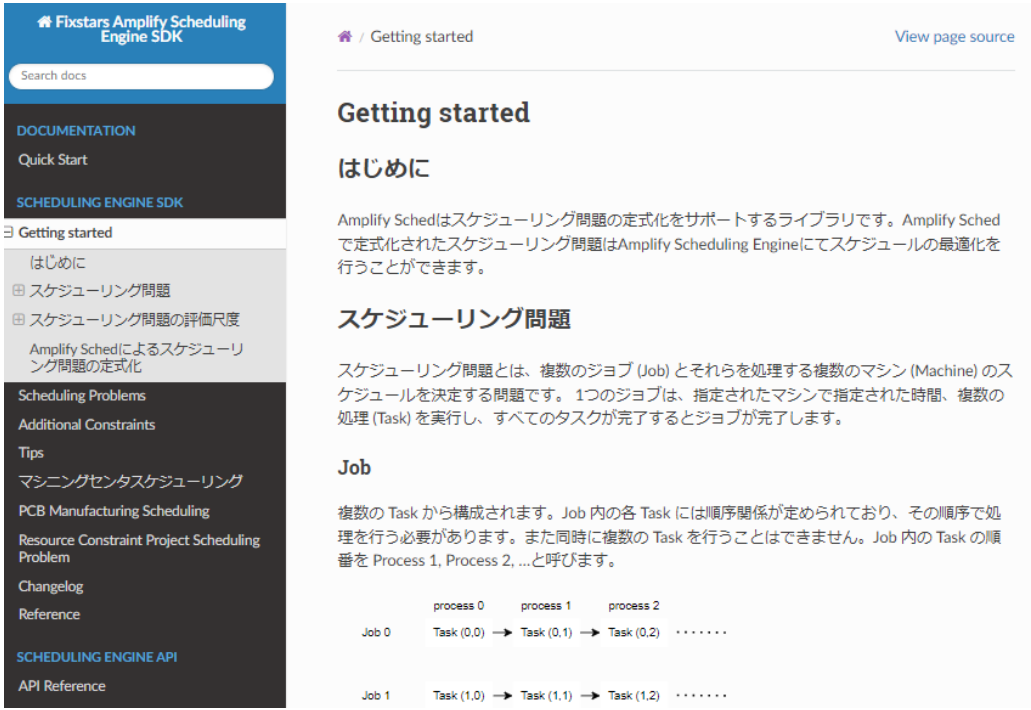
詳しく見る →

https://amplify.fixstars.com/ja/scheduling/resources#code



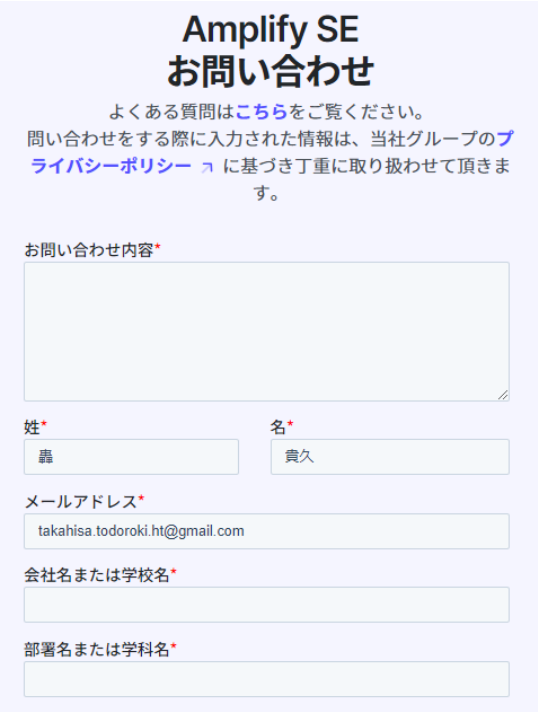
ドキュメント

https://amplify.fixstars.com/ja/docs/amplify-se/index.html

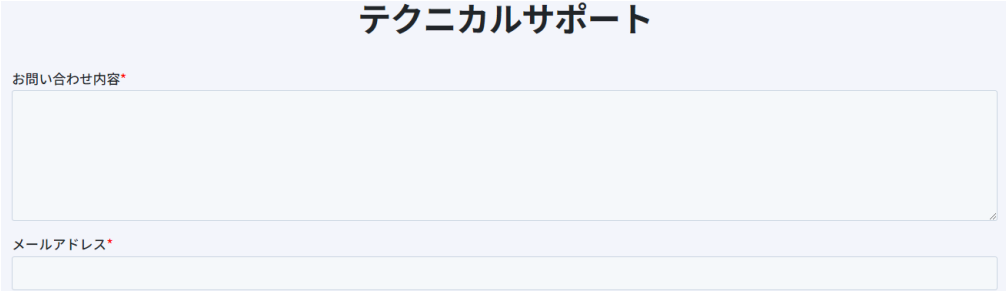


ご質問・ご不明点があればお問い合わせください！

https://amplify.fixstars.com/ja/scheduling/contact



https://amplify.fixstars.com/ja/user/contact



本セミナーのゴール

- 組合せ最適化問題を解くためのクラウドサービス「Fixstars Amplify」を知り、Fixstars Amplify Scheduling Engine を使ってスケジューリング最適化問題を解くための手法やプロセスを理解する

- ワークショップを通して、実際に Fixstars Amplify Scheduling Engine を使ってみることで、自社の業務への適用のイメージを掴む

今後のセミナーの予定

今後も定期的に無料セミナーを開催します！

2024/4/24 (水)
「生産計画最適化」
(Scheduling Engine)

- はじめに
- 会社紹介
- Fixstars Amplify Scheduling Engine のご紹介
- 生産計画最適化のワークショップ
- Wrap Up
  - 事例のご紹介
  - 今後の進め方
  - Q&A

2024/5/16 (仮)
「ブラックボックス最適化」
(Annealing Engine)

- はじめに
- 会社紹介
- Fixstars Amplifyの紹介
- ブラックボックス最適化のワークショップ
- Wrap Up
  - 事例のご紹介
  - 今後の進め方
  - Q&A

今回とほぼ同じ内容となる予定です

2024/5/30 (仮)
「生産計画最適化」
(Scheduling Engine)

- はじめに
- 会社紹介
- Fixstars Amplify Scheduling Engine のご紹介
- 生産計画最適化のワークショップ
- Wrap Up
  - 事例のご紹介
  - 今後の進め方
  - Q&A

ご質問・ご不明点がありましたら、お問い合わせフォームでご連絡下さい
https://amplify.fixstars.com/ja/contact

FIXSTARS Amplify

FIXSTARS Amplify

ご参加ありがとうございました！

アンケートのご協力お願いします