Denso - Transfer Plan Recommendation

Backend Architecture

Introduction

Transfer Plan Recommendation refers to a recommendation for relocating manufacturing of a product from one site to another. It could be either a subset of production steps or it could be the entire process. The goal here is to recommend possible transfer plans by optimizing a set of parameters while respecting a set of constraints. This document showcases the data requirements and the methodology followed for computing a transfer plan recommendation according to the given data.

Data Requirements

The data required to perform the transfer plan recommendation can be divided into two parts:

1. **Must Have** data - The smallest set that is required to compute **feasible, useful plans**.
2. **Optional** data - Data that improves **optimality, risk handling, and realism**.

***Must Have** data*

Product-level (one row per SKU)

- `product_id` (string) — unique key
- `monthly_demand` (pcs/month) — forecasted demand that must be met
- `unit_volume_or_weight` (optional proxy for storage) — pcs per pallet or m^3 per unit (if warehouse constraints used)
- `current_unit_cost` ($/unit) — current production unit cost (or best estimate)

Plant/Line-level (one row per candidate target plant/line)

- `plant_id` (string) — unique key
- `available_capacity` (pcs/month) — maximum producible units per month at the candidate site (nominal)
- `effective_oee` (0–1) or `availability_factor` — if available; otherwise assume 1.0 as default
- `unit_production_cost` ($/unit) — expected cost to produce that product at the target plant (if not available, use educated estimate or same as current unit cost)
- `transfer_fixed_cost` ($) — one-time cost to move/qualify the product to this plant (can be per product or per plant)

- `lead_time_to_start` (months) — time needed before production can start at that plant (can be 0)

Global / Transfer-level

- `budget_capital` ($) — optional: max one-time spend allowed (if you want cost cap)
- `transfer_deadline` (date or month) — must be respected (optional but useful)

Why these are must-have: with `demand`, `capacity`, `unit_costs`, and `transfer_costs`, you can formulate a basic constrained cost-minimization assignment (who produces what and where) and compute utilization and payback.

***Optional** data*

Product/process

- `cycle_time_sec` (sec/unit) and `required_machine_type` — for line matching and area planning
- `yield_rate` (%) — adjusts effective output required
- `special_compliance_flag` (boolean) — e.g., requires FDA/ISO/cleanroom
- `batch_size` or `lot_size` — affects setup and warehouse impacts

Plant/line

- `available_area_m2` and `area_required_per_product_m2` — physical fit checks
- `labor_skill_level` / `training_days_required` — for ramp planning and cost of training
- `warehouse_capacity_pallets` and `pallets_per_unit` — for storage constraints
- `interplant_transport_cost_per_unit` and `lead_time_days` — logistics cost/time
- `max_utilization_target` (%) — operational policy to avoid >90% loading
- `setup_time_hours` and `changeover_cost` — for multi-SKU lines

Financial & timing

- `monthly_demand_variability` (stdev or min/max) — for safety stock decisions
- `discount_rate` — for NPV payback calculations
- `depreciation_or_writeoff` for moved/retired assets

Risk & operational

- `risk_score` per plant (0–1) — supply chain, political, reliability
- `probability_of_delay` and `delay_cost_per_day`

Each of the above parameters improves realism, constraints, and multi-objective tradeoffs.

Optimization Approach

The approach in generating transfer plans is as follows: A list of variables that are required to optimize and a list of constraints must be defined. For example, based on the above data, some of the variables that are to be optimized are:

1. **Assignment (binary)**

   $x_{p,t} \in \{0,1\}$ = 1 if product *p* is produced at target plant *t* (full assignment).

   Optional fractional version if splitting production: $0 \leq v_{p,t} \leq D_p$.
2. **Volume allocation (continuous)**

   $v_{p,t}$ = units of product *p* produced at plant *t* (pcs/month).

   Enforces $\sum_t v_{p,t} = D_p$.
3. **Start time / scheduling (integer / discrete)**

   $s_{p,t}$ = start month when plant *t* begins production of *p* (affects lead time and inventory).
4. **Inventory buffer / safety stock (continuous)**

   $inv_{p,t,m}$ = inventory of *p* at plant *t* in month *m* during transfer.
5. **Utilization / resource usage (derived)**

   $u_t = \frac{\sum_p v_{p,t}}{Cap_t \times OEE_t}$ — used in constraints or objectives.
6. **Binary transfer decision (optional)**

   $y_{p,t}$ = 1 if you pay the one-time transfer_fixed_cost to enable production of *p* at *t*.

The objective functions to be followed during optimization are (**Choose any one**):

- **Minimize total cost**

  $$\min \sum_{p,t} \left(v_{p,t} \cdot c_{unit,p,t} + y_{p,t} \cdot C_{trans,p,t} + v_{p,t} \cdot c_{ship,p,t} \right)$$
- **Minimize time-to-full-production / downtime**
  Minimize sum of ramp delays or missed-delivery penalties.
- **Balance utilization / maximize capacity balance**

  Minimize variance of u_t across plants or maximize OEE-weighted throughput.

- **Multi-objective**

  Weighted sum: $\min w_1 \cdot Cost + w_2 \cdot Delay + w_3 \cdot Risk$. Or compute Pareto front

The constraints to be followed are:

- **Demand satisfaction:** $\sum_t v_{p,t} = D_p \quad \forall p$.
- **Capacity:** $\sum_p v_{p,t} \leq Cap_t \times OEE_t \quad \forall t$.
- **Activation:** $v_{p,t} \leq y_{p,t} \cdot Cap_t$ (only produce if transfer paid).
- **Warehouse:** Inventory and storage: $\sum_p inv_{p,t,m} \leq WHCap_t$.
- **Lead time / schedule:** If $s_{p,t}$ + lead_time > deadline then infeasible or penalize.
- **Budget (optional):** $\sum_{p,t} y_{p,t} \cdot C_{trans,p,t} \leq Budget$

To solve the above optimization with constraints problem, the following algorithms or approaches can be followed:

- **Mixed-Integer Linear Programming (MILP) -** (*Recommended*)
  - Suited because of binary assignment decisions (move or not [**Partial moving not possible**]) + costs + capacities and objectives are linear (cost minimization, capacity balancing).
  - **Libraries (Python):** Pyomo, GurobiPy(Python wrapper for a C library)
- **Linear Programming (LP) —** *If fractional splitting is required.*
- **Constraint Programming (CP)** — *If scheduling / sequencing is primary*