Optimal Workforce Scheduling in Care Services

Overview

Workforce scheduling in care services (such as in-home aged care or disability support) is a complex optimisation problem. It must balance the needs and constraints of both care providers and clients, including skills, availability, continuity of care, preferences, regulatory requirements, and travel times.

Common Approaches

- **Mathematical Optimisation:** Linear, integer, and mixed-integer programming are widely used to model and solve scheduling problems with constraints like shift coverage, maximum hours, and skill matching.
- **Constraint Programming:** Effective for handling complex rules (e.g., legal regulations, preferences, continuity of care).
- **Meta-heuristics:** Methods such as genetic algorithms, simulated annealing, and tabu search are applied to large, real-world problems where exact methods are computationally expensive.
- **Multi-objective Optimisation:** Many studies focus on balancing multiple goals, such as minimising costs, maximising continuity of care, and improving staff satisfaction.

Key Constraints and Rules

- Provider qualifications and skills (matching to client needs)
- Provider and client availability and preferences
- Regulatory requirements (rest breaks, maximum hours, etc.)
- Continuity of care (same provider for same client)
- Travel time and geographic constraints
- Emergency and last-minute changes

Mathematical Formulation

Scheduling care providers and clients is a bit like solving a big puzzle: you want to make sure every client gets the care they need, at the right time, from someone with the right skills, while also making sure providers aren't double-booked or overworked. To help with this, we use a structured approach that lists out everyone's needs, skills, and available times, and then finds the best way to match providers to clients across the schedule. This mathematical framework helps ensure the schedule is fair, efficient, and meets all the important rules and preferences.

Sets and Indices

- P: Set of Care Providers (indexed by p)
- C: Set of Clients (indexed by c)
- T: Set of Time Slots (indexed by t), each with a start time $start(t)$
- S: Set of Skills/Care Types (indexed by s)
- ℓ: Fixed slot length (e.g., 30 minutes)

Indicator Functions

- $I_{avail}(p, t)$: 1 if provider p is available at time slot t, 0 otherwise
- $I_{need}(c, s, t)$: 1 if client c needs care type s at time slot t, 0 otherwise
- $I_{skill}(p, s)$: 1 if provider p can deliver care type s, 0 otherwise

Parameters

- H_p: Maximum working hours for provider p in the planning period (e.g., per day)
- $C_{p,c,s,t}$: Cost (e.g., dollars per slot) for assigning provider p to client c for care type s at time slot t
- $\pi_{c,p}$: Preference score of client c for provider p (higher is better)

Decision Variable

- $x_{p,c,s,t} \in \{0, 1\}$: 1 if provider p is assigned to client c for care type s at slot t, 0 otherwise

Objective Function

The objective function is like the "goal" or "scorecard" for the scheduling system—it helps the system decide what makes one schedule better than another. In this context, the objective is to create a schedule that keeps costs as low as possible, matches clients with their preferred care providers when possible, and ensures everyone's needs and limits are respected. By setting up this goal, the system can automatically find the schedule that best balances efficiency, fairness, and satisfaction for both clients and care providers.

$$min \sum_{p \in P} \sum_{c \in C} \sum_{s \in S} \sum_{t \in T} \left(C_{p,c,s,t} \cdot x_{p,c,s,t} - \lambda \pi_{c,p} \cdot x_{p,c,s,t} \right)$$

where λ balances the costs against the preferences.

Constraints

Constraints are the "rules of the road" for the scheduling system—they make sure the schedule is safe, fair, and practical for everyone involved. These rules ensure, for example, that care providers are only assigned when they're available and have the right skills, that clients get the care they need at the right times, and that no one is double-booked or overworked. By following these constraints, the system creates a schedule that works smoothly for both clients and care providers, while also meeting important legal and quality standards.

1. **Provider Availability and Skill:**

$$x_{p,c,s,t} \leq I_{avail}(p,t) \cdot I_{skill}(p,s)$$

2. **Client Care Need Coverage:**

$$\sum_{p \in P} x_{p,c,s,t} \geq I_{need}(c,s,t) \; \forall c,s,t$$

3. **Provider Cannot Double Book:**

$$\sum_{c \in C} \sum_{s \in S} x_{p,c,s,t} \leq 1 \; \forall p,t$$

4. **Provider Maximum Working Hours:**

$$\sum_{c \in C} \sum_{s \in S} \sum_{t \in T} x_{p,c,s,t} \cdot \ell \leq H_p \; \forall p$$

User Interface (UI) Considerations

The user interface is how care providers and clients interact with the scheduling system—think of it as the “dashboard” or “control panel” for managing schedules. A good UI makes it easy for everyone to see their upcoming appointments, update their availability, request changes, and communicate important information. By focusing on clear calendars, simple forms, and helpful notifications, the system helps users quickly understand their schedules and make any needed adjustments, making the whole process smoother and less stressful for both staff and clients.

Care Provider App

- **Availability Calendar:** Select, edit, and submit available time slots (with recurring and holiday/leave options).
- **Shift Assignments:** View upcoming assigned shifts/visits, with client and care details.
- **Notifications:** For new assignments, changes, or messages.
- **Profile Management:** Update skills, preferences, and personal info.
- **Leave Management:** Submit/view leave periods.

Client App

- **Care Request Calendar:** Specify care needs, durations, and preferred providers.
- **Booking Overview:** See confirmed and pending bookings.
- **Notifications:** Reminders and updates.
- **Preference Setting:** Indicate preferred providers.

Current Trends in Workforce Scheduling Apps

- **AI-Driven Optimisation:** Use of AI/ML for matching, forecasting, and adapting to changes.
- **Mobile-First Access:** Scheduling and management via mobile apps.
- **Compliance & Credential Tracking:** Automated tracking of certifications, work hours, and regulations.
- **Integrated Communication:** Real-time messaging and alerts.
- **Self-Service:** Providers and clients manage their own schedules and preferences.
- **Analytics:** Dashboards and reporting for optimisation and compliance.

References

Academic & Technical

- Lieder, A., et al. (2015). "Task scheduling in long-term care facilities: A client-centered approach."
- Morse, A., et al. (2024). "Centralized Scheduling of Nursing Staff: A Rapid Review."
- Koruca, P., et al. (2023). "Development of a new personalized staff-scheduling method with a work-life balance perspective: case of a hospital."
- "Home health care routing and scheduling: A review," European Journal of Operational Research, 2015.
- Ma (2023) A Multi-Objective Scheduling and Routing Problem for Home Health Care Services via Brain Storm Optimization
- World Health Organization (2025). "Health and care workforce planning tools."

Industry & Implementation

- Novagems, Connecteam, Moris Media, Biarri, Softworks, OnPage, LogicBalls, ClickUp, etc. (see Appendix C for links)
- Open source: Staffjoy, OptaPlanner, Cal.com, Auto Shift Planner (see Appendix C for links)

Appendix A: Indicative Sample Data (Python-like dict)

```
care_providers = [
  {
    "id": "P001",
    "name": "Alex Smith",
    "skills": ["elderly_care", "medication_management"],
    "availability": [  # List of available slot IDs
      "T001", "T003", "T004"
    ],
    "max_hours": 8,  # Per day
    "preferences": ["C001"]  # Preferred clients
  },
  {
    "id": "P002",
    "name": "Jamie Lee",
    "skills": ["disability_support", "personal_care"],
    "availability": [
      "T002", "T004"
    ],
    "max_hours": 6,
    "preferences": ["C002"]
  }
]

clients = [
  {
    "id": "C001",
    "name": "Mary Johnson",
    "required_care": [  # List of care needs, each with skill and slots
      {"skill": "elderly_care", "slots": ["T001", "T003"]},
      # e.g., needs elderly care at 09:00 and 13:00
    ],
    "preferences": ["P001"]  # Preferred providers
  },
  {
    "id": "C002",
    "name": "Sam Wilson"
    "required_care": [
      {"skill": "disability_support", "slots": ["T002", "T004"]}
    ],
```

```
    "preferences": ["P002"]
  }
]

time_slots = [
  {"slot_id": "T001", "start": "2025-06-26 09:00", "length_mins": 30},
  {"slot_id": "T002", "start": "2025-06-26 10:00", "length_mins": 30},
  {"slot_id": "T003", "start": "2025-06-26 13:00", "length_mins": 30},
  {"slot_id": "T004", "start": "2025-06-26 14:00", "length_mins": 30}
]

assignments = [
  {
    "provider_id": "P001",
    "client_id": "C001",
    "skill": "elderly_care",
    "slot_id": "T001",
    "cost": 10
  },
  {
    "provider_id": "P002",
    "client_id": "C002",
    "skill": "disability_support",
    "slot_id": "T002",
    "cost": 12
  }
]
```

Appendix B: Approaching the Constrained Optimisation Problem

1. Model the Problem

- Define sets, parameters, and indicator functions as above.
- Encode all operational, regulatory, and preference constraints.
- Model appointments as sequences of consecutive slots.

2. Solution Approaches

- **Exact Methods:** Mixed-Integer Programming (MIP), Constraint Programming (CP) for small/medium problems.
- **Heuristics & Meta-heuristics:** Genetic algorithms, simulated annealing, tabu search for large/real-time problems.
- **Decomposition:** Break into sub-problems (e.g., slot assignment, then provider-client matching).
- **Rolling Horizon:** Solve for a short window, update as new data arrives.
- **AI/ML Augmentation:** Forecast demand, predict cancellations, recommend assignments.

3. Implementation Tips

- Start with a simple feasible solution, then iteratively improve.
- Ensure data (availability, skills, needs) is up to date.
- Test with real and synthetic data for robustness.
- Incorporate user feedback for continuous improvement.

For further details, see references and vendor documentation.

Appendix C: Possible Vendor Solutions

Open Source

- **OptaPlanner:** Advanced constraint solver for rostering and scheduling (OptaPlanner)
- **Staffjoy:** Open-source, self-hosted scheduling software (Staffjoy)
- **Cal.com:** Open-source appointment scheduling (Cal.com)
- **Auto Shift Planner:** Heuristic-based shift scheduling (Auto Shift Planner)

Commercial

- **Skedulo, Allocate, AlayaCare, Deputy, Rotageek, Novagems, ShiftCare, Biarri, QGenda, Smartlinx, AMiON** (see vendor websites for details).

Appendix D: Vendor Decision Matrix - Care Workforce Scheduling Solutions

Criteria	Weight	Vendor A	Vendor B	Notes/Comments
Core Scheduling Functionality	20%			Shift/appointment, multi-slot, recurring, split shifts
Healthcare/Care-Specific Features	10%			Compliance, skill matching, credential tracking
User Experience (UX/UI)	10%			Intuitive for carers, clients, admins
Mobile Capabilities	10%			Native apps, responsive design
Integration & API	10%			HRIS, payroll, time-tracking, SSO
Scalability & Performance	5%			Handles your org's size, future growth
Customization & Flexibility	5%			Custom fields, workflows, terminology
Compliance & Regulatory	10%			Labor laws, union rules, audit trails
Analytics & Reporting	5%			Dashboards, export, compliance reports
Support & Training	5%			Onboarding, documentation, live help
Implementation & Onboarding	5%			Time, resources, migration tools
Cost & Licensing	5%			Transparent pricing, TCO, ROI
Vendor Reputation & Roadmap	5%			References, innovation, updates