



INSTITUT FÜR INFORMATIK

Seminar Complex Scheduling

Projektplanung mit partiell erneuerbaren Ressourcen

Jan Elseberg

Wintersemester 2007/08

Diese Arbeit richtet sich hauptsächlich nach den Artikeln:
„Allocation of partially renewable resources: Concept, capabilities, and applications“ von Schirmer und Drexl [1]
und „Project Scheduling Under Partially Renewable Resource Constraints“ von Böttcher, Drexl, Kolisch und Salewski [3].

Inhaltsverzeichnis

1	Einleitung	1
2	Erweiterung des RCPSP	2
2.1	Motivation	2
2.2	Partiell erneuerbare Ressourcen	3
2.2.1	Beispiel:	4
2.3	Logische Relationen	5
3	Lösungsalgorithmen	9
3.1	Branch-and-Bound	9
3.2	Greedy Randomized Adaptive Search Procedure	12
4	Ergebnisse	15
4.1	ProGen-Erweiterung	15
4.2	Branch-and-Bound	15
4.3	Prioritätsregeln	16
5	Zusammenfassung	18

Kapitel 1

Einleitung

Die intelligente Planung komplexer Projekte, sowohl für wirtschaftliche wie wissenschaftliche Anwendungen, erfordert mächtige und eindeutige Problemformulierungen. Von besonderem Interesse ist hier die effiziente Nutzung von Arbeitskräften, Geld und anderen knappen Ressourcen über der Zeit.

Ein zentrales Problem ist das so genannte resource-constrained project scheduling problem (RCPSP). Das RCPSP macht es möglich, sehr allgemeine Bedingungen für Projektpläne zu definieren, allerdings ist es dennoch nicht in der Lage alle Konzepte zu modellieren. In der traditionellen Formulierung des RCPSP sind Ressourcen entweder über die gesamte Zeitspanne (nicht erneuerbare Ressourcen), oder über atomaren Zeitperioden (erneuerbare Ressourcen) des Plans beschränkt.

Um Bedingungen zu formulieren, bei denen über beliebigen zusammenhängenden und unzusammenhängenden Zeiten Einschränkungen gelten, wird das RCPSP um partiell erneuerbare Ressourcen erweitert, so dass eine neue Problemformulierung, das RCPSP/ π entsteht. Darüber hinaus können mit dem neuen Ressourcen-Konzept auf einfache Weise logische Relationen formuliert werden, was die Abstraktion konkreter Szenarien vereinfacht [1].

Die Erweiterung um partiell erneuerbare Ressourcen erfordert die Neuformulierung beziehungsweise den Ausbau von Lösungstechniken für das RCPSP. Nur wenige auf das RCPSP spezialisierte Verfahren sind dabei überhaupt übertragbar. Ein exakter Branch-and-Bound Algorithmus und ein heuristischer serieller Planungsalgorithmus, werden auf ihre Effizienz hin an generierten Beispieldaten untersucht [3].

Diese Arbeit konzentriert sich zunächst in Kapitel 2 auf die Erweiterung des RCPSPs. In Kapitel 2.3 werden Modellierungsfähigkeiten der partiell erneuerbaren Ressourcen diskutiert. Anschließend (Kap. 3) werden einige Lösungsalgorithmen vorgestellt und diese in Kapitel 4 evaluiert.

Kapitel 2

Erweiterung des RCPSP

In diesem Abschnitt wird zunächst die Einführung des neuen Ressourcen-Konzeptes an einem Beispiel motiviert und die Formulierung für das um partiell erneuerbare Ressourcen erweiterte RCPSP/ π vorgestellt. Darüber hinaus werden Richtlinien für die Modellierung logischer Relationen formuliert, welche das Modellierungspotential des RCPSP/ π demonstrieren.

Alle folgenden Abschnitte gehen davon aus, dass der Leser bereits über eingehende Kenntnisse über das RCPSP verfügt. Die verwendete Terminologie findet sich in Tabelle 2.1, auf eine genaue Definition des RCPSP ist hier jedoch verzichtet worden, für nähere Betrachtungen sei daher auf Brucker und Knust [5] verwiesen.

2.1 Motivation

Das nachfolgende Beispiel, angelegt an Schirmer und Drexl [1], soll die Restriktionen des RCPSP demonstrieren.

Parameter	Bedeutung
J	Menge der Aktivitäten mit $J = J $
d_j	Dauer der Aktivität j
$k_{jp}^\rho, (k_{jp}^\nu)$	Bedarf der Aktivität j von der (nicht) erneuerbaren Ressource p
K^ρ	Menge der erneuerbaren Ressourcen mit $K^\rho = K^\rho $
K_{pt}^ρ	Verfügbarkeit der erneuerbaren Ressource p zum Zeitpunkt t
K^ν	Menge der nicht erneuerbaren Ressourcen mit $J = K^\nu $
K_{pt}^ν	Verfügbarkeit der nicht erneuerbaren Ressource p
T	Zeit-Horizont mit $T = T $

Tabelle 2.1: Definition der wichtigsten Modell Parameter des RCPSP.

Für eine Gaststätte soll ein Arbeitsplan über eine Woche hinaus aufgestellt werden. Jeder Tag sei dabei in eine Früh- und eine Spätschicht eingeteilt. Der Planungshorizont T dieses Problems ließe sich also auf 14 beschränken. Die Gaststätte beschäftigt zwei Klassen von Arbeitnehmern: Vollzeitangestellte und Teilzeitangestellte.

Beide haben eine wöchentliche maximale Arbeitszeit; die Vollzeitangestellten arbeiten 4 Tage, die Teilzeitangestellten lediglich 2,5 Tage. Diese Bedingung lässt sich bereits durch das alte Konzept der nicht-erneuerbaren Ressourcen modellieren, indem jedem Arbeiter eine Ressource p mit einem Bedarf $k_p^\nu = 1$ und Kapazität $K_p^\nu = 8$ beziehungsweise $K_p^\nu = 5$ zugeordnet wird. Dies begrenzt die maximale Arbeitszeit über den gesamten Zeithorizont.

Um zu verhindern, dass ein Angestellter in einer Schicht mehrmals eingeplant wird, eignen sich die erneuerbaren Ressourcen. Jeder Arbeiter habe einen Bedarf $k_{p'}^\rho = 1$ der Ressource p' , die über jeder Zeitperiode mit $K_{p'}^\rho = 1$ beschränkt ist. Erlaubt man ein zeitabhängiges Ressourcenprofil, so ließe sich zusätzlich die Arbeitszeit des Vollzeitangestellten auf die Werkzeuge beschränken, indem man die Kapazität $K_{p't}^\rho$ an den letzten 4 Schichten ($t = 11, \dots, 14$) auf 0 begrenzt.

Auf diese Art und Weise lassen sich mit den gewöhnlichen Ressourcen bereits einige Konzepte modellieren. An den folgenden Anforderungen für den Wochenplan scheitert das herkömmliche RCPSP jedoch.

Die Angestellten der Gaststätte haben sich in einer Gewerkschaft organisiert und verlangen nun, dass kein Arbeiter mehr als einen Wochenendtag pro Woche zu arbeiten hat, also dass der Angestellte entweder Samstag oder Sonntag frei hat. Da nun das Einplanen einer Aktivität am Samstag/Sonntag einen Effekt auf Sonntag/Samstag haben soll, wie bei den nicht-erneuerbaren Ressourcen, dies aber nicht für alle Zeitperioden gleichermaßen gilt, sind weder erneuerbare noch nicht-erneuerbare Ressourcen in der Lage die Forderungen der Gewerkschaft zu erfüllen.

Um die Angestellten dennoch bei Laune zu halten, entscheidet die Geschäftsführung, dass jede Woche wenigstens einer der drei Manager eine Schicht am Wochenende übernimmt. Leider ist auch diese, eher als logische Relation zwischen Planungsentscheidungen zu sehende Anforderung unmöglich mit dem RCPSP zu modellieren.

2.2 Partiiell erneuerbare Ressourcen

Zur Bewältigung der oben präsentierten Probleme, wird nun das Ressourcen beschränkte Projekt Planungsproblem unter partiiell erneuerbaren Ressourcen (RCPSP/ π) nach Schirmer und Drexel [1] definiert.

Die meisten der bereits bekannten Modellparameter des RCPSPs bleiben unverändert. Noch immer müssen J Aktivitäten mit $J = \{1, \dots, J\}$ innerhalb von T Zeitperioden mit $T = \{1, \dots, T\}$ bearbeitet werden. Jede Aktivität, von nun an mit j indiziert, muss für d_j zusammenhängende Zeitperioden eingeplant werden. In jeder Zeitperiode werden k_{jp} Einheiten der partiiell erneuerbaren Ressource p von der Aktivität j verbraucht. Dieser Verbrauch ist kumulativ und der Gesamtverbrauch aller zum Zeitpunkt t ausgeführten Aktivitäten darf die Kapazität der Ressource

Parameter	Bedeutung
J	Menge der Aktivitäten mit $J = J $
d_j	Dauer der Aktivität j
k_{jp}	Bedarf der Aktivität j von der partiell erneuerbaren Ressource p
P	Menge der partiell erneuerbaren Ressourcen mit $P = P $
K_p	Verfügbarkeit der partiell erneuerbaren Ressource p
T	Zeit-Horizont mit $T = T $
$\Pi_p \subseteq T$	Teilmengen des Zeithorizontes über denen die Ressource p beschränkt ist.

Tabelle 2.2: Definition der wichtigsten Modell Parameter des RCPSP/ π .

p nicht überschreiten. Die partielle Ordnung \prec auf den Aktivitäten gibt die Präzedenzrelationen wieder.

Die klassischen Ressourcen des RCPSP werden vollständig von P partiell erneuerbaren Ressourcen mit $P = \{1, \dots, P\}$ ersetzt. Jede Ressource p ist über genau einer durch $\Pi_p \subseteq T$ gegebenen Teilmenge mit einer Kapazität von K_p beschränkt. Dies ist die Normalisierung einer Formalisierung, bei der Ressourcen über mehreren Teilmengen mit verschiedenen Kapazitäten beschränkt sind. Dies bedeutet jedoch keinerlei Einschränkung für die Mächtigkeit der Modellierung sondern ermöglicht lediglich einfachere Formulierungen. Sollte eine Zeitperiode nicht durch eine der Teilmengen abgedeckt sein, so gilt für diese Periode keine Ressourcenbeschränkung.

Ziel ist es nun, für jede Aktivität j eine Startzeit zu finden (einen Plan), so dass sichergestellt ist, dass für jede Ressource p der Gesamtverbrauch aller Aktivitäten von p innerhalb der dazugehörigen Teilmenge unterhalb der Kapazität K_p liegt, alle Präzedenzrelationen befolgt werden und die Länge des Plans minimiert wird.

Um einige Betrachtungen zu vereinfachen werden nun zusätzlich zu den Modell-Parametern (siehe Tabelle 2.2) einige Werte hergeleitet. Häufig ist es sinnvoll so genannte *dummy* Aktivitäten einzuführen, die weder Ressourcen noch Zeit verbrauchen. Gelte also für die Aktivität 1, dass sie vor allen anderen Aktivitäten ausgeführt wird und für J dass sie nach allen anderen ausgeführt wird. Zusätzlich sei $d_1 = d_J = 0$ und $k_{1p} = k_{Jp} = 0$ für alle p .

Sei $Pred_j$ die Menge aller direkten Vorgänger von j . Die Werte EFT_j und LFT_j seien die frühest beziehungsweise spätest mögliche Fertigstellungszeit der Aktivität j . Dementsprechend bezeichnen EST_j und LST_j die frühest und spätest möglichen Startzeiten von j . Diese Werte können durch Vorwärts- und Rückwärts-Rekursion durch den Präzedenzgraphen ermittelt werden.

2.2.1 Beispiel:

Um die Vorteile des RCPSP/ π 's zu verdeutlichen, wird das Beispiel aus Abschnitt 2.1 vollständig mit partiell erneuerbaren Ressourcen modelliert. Sei wieder eine Woche mit $T = 14$ Schichten sowohl für Vollzeit- wie Teilzeitangestellte zu planen.

Um die Gesamtarbeitszeit zu beschränken wird für die Vollzeitangestellten jeweils eine Ressource p eingeführt mit $K_p = 8$ und $\Pi_p = \{1, \dots, 14\}$. Für die Teilzeitangestellten wird dementsprechend ein Ressource p' mit $K_{p'} = 5$ und $\Pi_{p'} = \{1, \dots, 14\}$ eingeführt. Dies emuliert korrekt die nicht-erneuerbaren Ressourcen und sorgt dafür, dass das Wochenpensum nicht überschritten wird.

Um auch die erneuerbaren Ressourcen des Beispiels umzusetzen, werden für jeden Arbeiter und jeden Zeitpunkt t eine partiell erneuerbare Ressource p_t benötigt, die über der entsprechenden Teilmenge $\Pi_{p_t} = \{t\}$ beschränkt ist. Für Vollzeitangestellte ist die Kapazität dieser Ressourcen jeweils 1, während die Ressourcen für Teilzeitangestellte an den Zeiten $t = 11, 12, 13, 14$ eine Kapazität von 0 und sonst eine Kapazität von 1 zugeteilt werden.

Zusätzlich lassen sich die Forderungen der Gewerkschaft durch 4 partiell erneuerbare Ressourcen pro Arbeiter erfüllen:

$$\begin{array}{ll} K_1 = 1 & \Pi_1 = \{11, 13\} \\ K_2 = 1 & \Pi_2 = \{11, 14\} \\ K_3 = 1 & \Pi_3 = \{12, 13\} \\ K_4 = 1 & \Pi_4 = \{12, 13\} \end{array}$$

Den Zeithorizont auf diese Weise aufzuteilen hat die Folge, dass sobald ein Arbeiter am Samstag bzw. Sonntag eingeteilt wird, keinerlei Kapazität von mindestens 2 Ressourcen am Sonntag bzw. Samstag übrig bleibt.

Zuletzt wird die Forderung der Geschäftsleitung durch eine partiell erneuerbare Ressource p mit Zeithorizont $\Pi_p = \{11, 12, 13, 14\}$ und negativer Kapazität $K_p = -1$ modelliert. Zusätzlich ist für die Manager ein negativer Ressourcenbedarf $k_p = -1$ erforderlich. Nun muss mindestens einer der Manager am Wochenende eingeteilt werden um dafür zu sorgen dass die Ressource eine Kapazität ≥ 0 erhält.

2.3 Logische Relationen

Partiell erneuerbare Ressourcen haben einen hohen Grad an Ausdrucksfähigkeit. Unter anderem lassen sich Randbedingungen modellieren, welche vom natürlichen Verständnis zunächst wenig mit Ressourcen zu tun haben. Darunter fallen häufig logische Relationen, die für die Planung von Aktivitäten innerhalb eines geschlossenen Zeitfensters zu gelten haben. Die nachfolgenden Richtlinien (nach [1]) zur Modellierung solcher Relationen, dienen somit der einfachen Modellierung realer Zusammenhänge.

Identität: Die logische Identität ist die kleinste Einheit der logischen Relationen, und ist gleichbedeutend mit der Aussage: Eine Aktivität wird innerhalb eines Zeitfensters bearbeitet.

Sei $T' = \{t_i, \dots, t_n\}$ ein zusammenhängendes Intervall im gesamten Zeithorizont, und $j \in J$ eine Aktivität mit $d_j \leq n$, so dass sichergestellt ist, dass j innerhalb von T' bearbeitet

werden kann. Dann lässt sich die Identität wie folgt mit einer partiell erneuerbaren Ressource formulieren:

$$\Pi_p = T \setminus T', \quad K_p = 0, \quad k_{jp} = 1$$

Da zu allen Zeitpunkten an denen die Aktivität nicht ausgeführt werden soll weniger Kapazität zur Verfügung steht als sie verbraucht, kann j nur an den nicht beschränkten Zeitpunkten eingeplant werden. Daher kann die Identität mit allen k_{jp} und K_p modelliert werden für die gilt: $k_{jp} > K_p$.

Sind negative Werte für Kapazität und Ressourcenbedarf erlaubt, so lässt sich eine kompaktere Darstellung finden, die nicht alle anderen Zeitperioden „verbietet“ sondern von der Aktivität verlangt die Ressource innerhalb von T' „aufzufüllen“.

$$\Pi_p = T', \quad K_p = -d_j, \quad k_{jp} = -1$$

Es genügt eine Kapazität K_p und ein Ressourcenbedarf k_{jp} mit $k_{jp}d_j > K_p$.

Negation: Das logische Gegenstück zu der Identität ist die Negation, also die Forderung, dass eine Aktivität j innerhalb einer Teilmenge T' des Zeithorizontes nicht bearbeitet wird. Im Gegensatz zu der logischen Identität kann auf den Zusammenhang von T' verzichtet werden. Eine partiell erneuerbare Ressource p mit

$$\Pi_p = T', \quad K_p = 0, \quad k_{jp} = 1,$$

sorgt für das Einhalten der Forderung, indem innerhalb des betreffenden Zeithorizonts T' weniger Kapazität zur Verfügung steht als die Aktivität benötigt. Daher genügt für die Negation jede Kombination von Kapazität K_p und Ressourcenbedarf k_{jp} für die gilt: $k_{jp} > K_p$.

Konjunktion: Die UND-Verknüpfung ist eine n -stellige Relation, die genau dann wahr ist wenn alle ihre Teilaussagen wahr ist. Sie ist demnach eine Erweiterung der logischen Identität, und repräsentiert die Forderung dass eine Menge von Aktivitäten innerhalb eines Zeitfensters bearbeitet werden.

Sei $T' = \{t_i, \dots, t_n\}$ erneut ein zusammenhängendes Zeitintervall, und J' eine Teilmenge aller Aktivitäten mit $d_j \leq n$, die alle innerhalb von T' bearbeitet werden sollen. Dann lässt sich auf zweierlei Arten eine partiell erneuerbare Ressource p definieren mit

$$\Pi_p = T \setminus T', \quad K_p = 0, \quad k_{jp} = 1 \quad \forall j \in J'$$

oder

$$\Pi_p = T', \quad K_p = - \sum_{j \in J'} d_j, \quad k_{jp} = -1 \quad \forall j \in J'.$$

Wie bei der Identität verbietet die erste Alternative Aktivitäten innerhalb des Zeitintervalls $T \setminus T'$ einzuplanen (gilt für alle $k_{jp} > K_p$), während die zweite Alternative erzwingt alle Aktivitäten innerhalb von T' einzuplanen. Verallgemeinert sind in diesem Fall alle K_p, k_{jp} ausreichend mit $k_{jp} \sum_{j \in J'} d_j = K_p < 0$.

Disjunktion: Die ODER-Verknüpfung ist eine n-stellige logische Relation, die genau dann wahr ist, wenn eine oder mehrere ihrer Teilaussagen wahr ist. Mit anderen Worten, sie verkörpert die Forderung, dass mindestens eine Aktivität innerhalb eines gewissen Zeitfensters bearbeitet wird.

Seien die Größen T' und J' wie zuvor definiert, und gelte zusätzlich $d_j = 1$ für alle $j \in J'$, dann lässt sich die Verknüpfung durch die partiell erneuerbare Ressource p mit

$$\Pi_p = T', \quad K_p = -1, \quad k_{jp} = -1 \quad \forall j \in J',$$

modellieren. Durch die negative Kapazität muss mindestens eine Aktivität in T' eingeplant werden um die Ressource „aufzufüllen“. Die Einheitslängen der Aktivitäten sind in diesem Fall notwendig, da sonst Aktivitäten nur teilweise in T' bearbeitet werden könnten ohne Ressourcenbegrenzungen zu verletzen.

Negierte Konjunktion Die NAND-Verknüpfung ist die n-stellige Relation, die immer dann wahr ist, wenn die UND-Verknüpfung nicht wahr ist und umgekehrt. Diese Relation gewährleistet also dass aus einer Menge von Aktivitäten mindestens eine nicht innerhalb eines bestimmten Zeitfensters bearbeitet wird.

Ist $T' = \{t_i, \dots, t_n\}$ zusammenhängend und J' eine Teilmenge aller Aktivitäten mit $d_j \leq n$ von denen nicht alle innerhalb von T' bearbeitet werden dürfen, so modelliert die partiell erneuerbare Ressource p mit folgenden Größen die NAND-Relation.

$$\Pi_p = T', \quad K_p = \sum_{j \in J'} d_j - 1, \quad k_{jp} = 1 \quad \forall j \in J'.$$

Falls alle Aktivitäten in T' bearbeitet würden, wäre ihr Gesamtverbrauch $\sum_{j \in J'} d_j$ der Ressource p grösser als dessen Kapazität K_p , so dass mindestens eine Aktivität nicht in T' eingeplant werden kann.

Negierte Disjunktion Die NOR-Verknüpfung ist die n-stellige Relation, die immer dann wahr ist, wenn die ODER-Verknüpfung nicht wahr ist und umgekehrt. Sie sichert also ab, dass aus einer Menge von Aktivitäten keine innerhalb eines bestimmten Zeitfensters bearbeitet wird, und ist somit eine Verallgemeinerung der Negation.

Dementsprechend sei $T' \subset T$ eine nicht notwendigerweise zusammenhängende Teilmenge des Zeithorizonts, und J' eine Menge von Aktivitäten von denen keine innerhalb von T' bearbeitet werden darf. Dieser Sachverhalt lässt sich mit der partiell erneuerbaren Ressource p mit

$$\Pi_p = T', \quad K_p = 0, \quad k_{jp} = 1, \quad \forall j \in J'$$

modellieren. Dies ist eine Erweiterung der Negation, so dass auch hier alle $k_{jp} > K_p$ ausreichen.

Exklusive Disjunktion Die exklusive Disjunktion (oder XOR-Verknüpfung) ist genau dann wahr, wenn genau eine ihrer Teilaussagen wahr ist. Dies lässt sich verallgemeinern zu der Bedingung, dass genau m Teilaussagen wahr sein müssen, also dass genau m Aktivitäten innerhalb einer bestimmten Zeitperiode eingeplant werden.

Für die korrekte Modellierung dieser Aussage, werden zwei partiell erneuerbare Ressourcen p und p' benötigt, welche die Bedingungen „mindestens m Aktivitäten“ und „höchstens m Aktivitäten“ garantieren. Es gelten daher für die Aktivitäten J' erneut die Einheitsdauern $d_j = 1$. Darüber hinaus sei $T' = \{t_i, \dots, t_n\}$ ein zusammenhängendes Zeitintervall in denen m der Aktivitäten von J' bearbeitet werden sollen.

- Ressource p , mindestens m Aktivitäten:

$$\Pi_p = T', \quad K_p = -m, \quad k_{jp} = -1 \quad \forall j \in J'.$$

So ist sichergestellt, dass wenigstens m Aktivitäten die Ressource p „auffüllen“. Um dies zu gewährleisten genügen alle K_p und k_{jp} mit $k_{jp} = K_p/m < 0$

- Ressource p' , höchstens m Aktivitäten:

$$\Pi_{p'} = T', \quad K_{p'} = m, \quad k_{jp'} = 1 \quad \forall j \in J'$$

Auf diese Weise können in dem Zeitraum T' maximal m Aktivitäten ausgeführt werden. Es reichen hierbei bereits K_p und k_{jp} mit $k_{jp} = K_p/m$ aus.

Somit kann eine Vielzahl von logischen Relationen auf einfache Weise mit partiell erneuerbaren Ressourcen modelliert werden, jedoch hat auch dieser Ansatz seine Grenzen.

Mit den oben formulierten Richtlinien ist es nicht möglich alle denkbaren logische Ausdrücke umzusetzen, da jede Verknüpfung der Richtlinien einer Komposition gleichkommt, d.h. man kann nur fordern, dass Bedingung 1 **und** Bedingung 2 erfüllt seien sollen. Um dennoch den gesamten Bereich der logische Aussagen abzudecken, müsste zusätzlich zu der Komposition der Bedingungen eine Negation möglich sein, was hier nicht der Fall ist.

Kapitel 3

Lösungsalgorithmen

Lösungsalgorithmen, die auf das RCPSP/ π spezialisiert sind, sind noch immer selten. Aufgrund dieses Mangels an geeigneten Algorithmen haben Böttcher et. al [3] zwei Lösungsverfahren entworfen, die auf bekannten RCPSP-Methoden aufbauen. Anfangs stellt dieses Kapitel den erweiterten exakten Branch-and-Bound Algorithmus basierend auf Präzedenzbäumen in Abschnitt 3.1 vor, im Anschluss daran wird in Abschnitt 3.2 der heuristische auf Prioritätsregeln basierende Planungsalgorithmus *greedy randomized adaptive search procedure* (GRASP) vorgestellt.

3.1 Branch-and-Bound

Einer der effektivsten exakten Algorithmen für das gewöhnliche RCPSP, das Branch-and-Bound Verfahren basierend auf Verspätungsalternativen von Demeulemeester et al. [2] ist nicht für das RCPSP/ π geeignet. Dies trifft, mit Ausnahme des auf Präzedenzbäumen basierenden Branch-and-Bound Verfahrens von [4] auf die meisten Branch-and-Bound Algorithmen zu.

Böttcher et. al [3] erweitern daher den Branch-and-Bound Algorithmus mit Präzedenzbäumen dahingehend, dass zusätzlich zu der Aktivität j die Startzeit t von j in jedem Knoten des Baumes gespeichert wird. Der Algorithmus führt also eine Tiefensuche in einem Präzedenzbaum der Tiefe J und mit der Wurzel 1 durch. In jeder Ebene des Baumes wird eine zusätzliche präzedenzzulässige Aktivität j am frühest möglichen, ressourcenzulässigen Zeitpunkt $t \in \{EST_j, \dots, LST_j\}$ zu dem bisherigen Teilplan hinzugefügt.

Ein Teilplan S sei definiert durch eine Folge von Tupeln (i, j_i, t_i) , mit $n \leq J$ Ebenen des Baumes. Jeder Ebene $i = 1, \dots, n$ ist eine Aktivität j_i und dessen Startzeit t_i zugeordnet. Sei SC_{jpt} der Verbrauch der Ressource p von Aktivität j , wenn diese zum Zeitpunkt t bearbeitet wird. Dann ist die übrig gebliebene Kapazität K_p^0 der Ressource p in einem Teilplan S gegeben durch:

$$K_p^0 = K_p - \sum_{i=1}^n SC_{j_i p t_i}.$$

Ein Teilplan S ist also solange zulässig wie $K_p^0 \geq 0$ für alle p gilt und alle Präzedenzrelationen erfüllt sind.

Der Algorithmus setzt immer zurück wenn es keine Erweiterung gibt, die zu einem zulässigen Teilplan führt oder wenn die bisherige obere Schranke \bar{T} des Makespans vom Teilplan überschritten wird. In diesem Fall hebt der Algorithmus die letzte Planungsentscheidung auf, inkrementiert aber zunächst nur die Startzeit der zuletzt gewählten Aktivität j . Eine andere Aktivität j' wird erst dann gewählt wenn keine Startzeiten von j verbleiben. Wann immer ein Teilplan erstellt wird, der alle Aktivitäten beinhaltet und dessen Bearbeitungszeit unter \bar{T} liegt, wird die obere Schranke \bar{T} aktualisiert. Der Algorithmus terminiert wenn alle möglichen Pläne enumeriert wurden, also wenn Aktivität 1 aus dem Teilplan entfernt wird.

Um den enormen Aufwand, den der Algorithmus bewältigen muss, einzuschränken, werden im folgenden Zulässigkeitschranken aufgestellt. Dies sind keine Schranken im eigentlichen Sinne, sondern stattdessen hinreichende Bedingungen um die Unzulässigkeit eines Teilplanes festzustellen und so Erweiterungsalternativen im Präzedenzbaum zu eliminieren.

Zulässigkeitschranke 1 Die Idee hinter dieser Schranke ist es den Bedarf an partiell erneuerbaren Ressourcen genauer abzuschätzen um eine weitere Unzulässigkeitsbedingung herzuleiten. Dabei wird in jeder Ebene n , für jede Ressource eine untere Schranke des Verbrauchs von allen Nachfolgern der Aktivität j_n errechnet. Sollte dieser minimale Verbrauch über der Kapazität der entsprechenden Ressource liegen, ist der Teilplan unzulässig.

Wenn die Aktivität j nicht früher als t gestartet wird, also im Intervall $[t, LST_j]$, ist ihr minimaler Verbrauch der Ressource p gegeben durch:

$$MC_{jpt} = \min\{SC_{jp\tau} | t \leq \tau \leq LST_j\}.$$

Offensichtlich gilt aufgrund des *min*-Ausdrucks:

$$MC_{jpt} \leq MC_{jpt+1},$$

für alle t . Sei $Succ_j$ die Menge aller direkten und indirekten Nachfolger von j , und LP_{jh} die Länge des längsten Pfades von j nach $h \in Succ_j$. Dann ist

$$EST'_h = \max\{EST_h, t_n + LP_{j_n, h}\}$$

eine untere Schranke für die früheste Startzeit der Aktivität h , wenn j_n auf Ebene n zum Zeitpunkt t_n eingeplant wurde.

Mit diesen Größen ist der minimale Verbrauch $MCI_{j_n p}$ der Ressource p aller Nachfolger von j_n gegeben durch:

$$MCI_{j_n p} = \sum_{h \in Succ_j} MC_{hpEST'_h}.$$

Für einen zulässigen Teilplan muss also die zur Verfügung stehende Kapazität K_p^0 jeder Ressource p den minimalen Verbrauch MCI_{jnp} übersteigen:

$$K_p^0 \geq MCI_{jnp}.$$

Zulässigkeitschranke 2 Bei der zweiten Zulässigkeitschranke werden aus nicht einplanbaren Aktivitäten so genannte Unzulässigkeitsinformationen gewonnen, die von allen noch vom Algorithmus erstellten Teilplänen widerlegt werden müssen. Kann ein Teilplan diese Bedingung nicht erfüllen, so wird er nicht weiter berücksichtigt.

Wann immer eine Aktivität j innerhalb ihres Zeitfensters $\{EST_j, \dots, LST_j\}$ nicht eingeplant werden kann, wird der folgende Vektor errechnet und in einer LIFO-Struktur abgelegt.

$$\varsigma_j = (j, \tau_j, MC_{j1\tau_j}, \dots, MC_{jp\tau_j}, (p_{\tau_j}, \Delta_{\tau_j}), \dots, (p_{LST_j}, \Delta_{LST_j}))$$

Hierbei ist τ_j der frühest mögliche Zeitpunkt an dem j im Teilplan einplanbar ist, wenn nur Präzedenzrelationen beachtet werden. Ebenfalls enthält der Vektor den minimalen Ressourcenbedarf $MC_{jp\tau_j}$ von j für alle partiell erneuerbaren Ressourcen. Zusätzlich wird für jeden Zeitpunkt $t \in \{\tau_j, LST_j\}$ jeweils eine Ressource p_t gespeichert, für die der Verbrauch von j die übrig gebliebene Kapazität um Δ_{t_j} Einheiten übersteigt.

Zur gesamten Laufzeit des Algorithmus wird höchstens eine Unzulässigkeitsinformation ς_j für jede Aktivität j gespeichert. Wann immer ein weiterer Teilplan expandiert werden soll, muss dieser alle ς_j aufheben können. Dies kann auf eine von zwei Weisen geschehen.

- Die Aktivität j im aktuellen Teilplan kann vor dem Zeitpunkt τ_j eingeplant werden. In diesem Fall ist es möglich, dass ein ressourcenzulässiger Plan existiert.
- Es gibt mindestens eine Ressource p_t , für die genügend Kapazität zum Zeitpunkt t übrig ist:

$$K_{p_t}^0 \geq \Delta_t.$$

Weiterhin muss für alle Ressourcen p gelten:

$$K_p^0 \geq MC_{jpt}$$

Auch dann kann noch ein zulässiger Plan entstehen.

3.2 Greedy Randomized Adaptive Search Procedure

Um reale Problem mit einer Vielzahl an Aktivitäten und Ressourcen zu handhaben, ist es oft sinnvoll auf die Optimalität des Lösungsalgorithmus zu verzichten. Eine Untermenge dieser heuristischen Algorithmen sind die Prioritätsregel-basierten Planungsalgorithmen, bei denen in Schritten ein partieller Plan durch eine oder mehrere Aktivitäten ergänzt wird. Die in jedem Schritt getroffene Auswahl wird durch eine Prioritätsregel vorgeschrieben, und kann sowohl deterministisch als auch indeterministisch, mit einer Wahrscheinlichkeitsverteilung basierend auf der Prioritätsregel, geschehen.

Der serielle Planungsalgorithmus von Drexl und Kolisch [6] wurde von Böttcher et al. [3] modifiziert um partiell erneuerbare Ressourcen zu handhaben. Die daraus entstandene „greedy randomized adaptive search“ Prozedur (GRASP) wählt einzuplanende Aktivitäten indeterministisch (randomized) nach einer von vielen möglichen Prioritätsregeln (adaptive) aus, und kann daher häufiger durchgeführt werden um verschiedene und evtl. bessere Pläne zu generieren.

Der Algorithmus konstruiert in höchstens J Schritten einen Plan, indem bei jedem Schritt i genau eine Aktivität j_i zum Zeitpunkt t_i dem Teilplan S hinzugefügt wird. In jedem Schritt werden zwei disjunkte Mengen C und D errechnet. In C befinden sich alle eingeplanten Aktivitäten, während D alle einplanbaren Aktivitäten j und ihre möglichen Startzeitpunkte t enthält. D ist gegeben durch:

$$\begin{aligned}
 D := & \{(j, t) \mid (j \notin C) && j \text{ ist noch nicht eingeplant} \\
 & \wedge (\forall h \in \text{Pred}_j : h \in C) && j \text{ hat keine nicht eingeplanten Vorgänger} \\
 & \wedge (t \in \{EST'_j, \dots, LST_j\}) && t \text{ ist innerhalb des einplanbaren Zeitintervalls von } j \\
 & \wedge (\forall p \in P : K_p^0 \geq SC_{jpt} + MCI_{jpt}) && j, t \text{ ist ressourcenzulässig}
 \end{aligned}$$

Der Algorithmus terminiert entweder erfolgreich nach J Schritten oder vorher, falls keine weiteren Aktivitäten mehr eingeplant werden können, mit $D = \emptyset$.

Auswahlverfahren: Um aus der Menge D eine Auswahl (j, t) zu treffen, wird eine Wahrscheinlichkeitsverteilung ψ mit

$$\psi : (j, t) \in D \rightarrow [0, 1]$$

benötigt. Definieren wir zunächst eine Prioritätsregel ω als eine Funktion, die jedem Tupel $(j, t) \in D$ einen Prioritätswert $\omega_{jt} \geq 0$ zuordnet und zusätzlich ein Ziel $O = \min, \max$, das festlegt ob kleine oder große Prioritätswerte favorisiert werden sollen. Nun lässt sich ein Hilfwert, die *Reue* ρ_{jt} festlegen:

$$\rho_{jt} := \begin{cases} \max\{\omega_{i,\tau} \mid (i, \tau) \in D\} - \omega_{jt} & , \text{ falls } O = \min \\ \omega_{jt} - \min\{\omega_{i,\tau} \mid (i, \tau) \in D\} & , \text{ falls } O = \max \end{cases}$$

Bezeichnung	Bedeutung	Modus
MINEFT	Minimale früheste Beendigungszeit	$O = \min$
MINLFT	Minimale späteste Beendigungszeit	$O = \min$
MINSLK	Minimaler Slack	$O = \min$
MTSUCC	Maximale Anzahl an Nachfolger	$O = \max$

Tabelle 3.1: Übersicht über klassische Prioritätsregeln für das RCPSP.

Die Reue ρ_{jt} bezeichnet für jedes Tupel (j, t) also den Prioritätsunterschied zu dem schlechtesten Tupel. Nun lässt sich die Wahrscheinlichkeitsverteilung ψ wie folgt definieren:

$$\psi(j, t) := \frac{(\rho_{jt} + 1)^\alpha}{\sum_{(i, \tau) \in D} (\rho_{i\tau} + 1)^\alpha}.$$

Die Konstante 1 sorgt dafür, dass auch dem schlechtesten Tupel eine Wahrscheinlichkeit grösser 0 zugeordnet wird. Der Parameter α stellt ein mit welcher Tendenz gute Tupel bevorzugt werden. Bei $\alpha = \infty$ trifft die Auswahl deterministisch immer das Tupel mit Reue $\rho_{jt} = 0$, bei $\alpha = 0$ ist ψ eine Gleichverteilung.

Statische Prioritätsregeln Die für das RCPSP bekannten statischen Prioritätsregeln lassen sich ohne Änderungen übernehmen. Für eine Übersicht über die verwendeten Bezeichnungen siehe Tabelle 3.1

Der maximale statische Ressourcenbedarf einer Aktivität j sei gegeben durch

$$SRU_j = \sum_{p \in P} k_{jp}.$$

So lassen sich mit $O = \max$ bzw. $O = \min$ und $\omega_{jt} = SRU_j$ die Prioritätsregel des maximalen bzw. minimalen statischen Ressourcenbedarfs MAXSRU bzw. MINSRU definieren.

SRU_j hängt weder von der Startzeit t , noch vom bisher generierten Teilplan S oder von der noch zur Verfügung stehenden Kapazität der Ressourcen ab. Daher lassen sich die Prioritätsregeln des zeitabhängigen statischen Ressourcenbedarfs MAXTRU und MINTRU mit $\omega_{jt} = TRU_{jt}$, und $O = \max, \min$ entsprechend, definieren:

$$TRU_{jt} = \sum_{p \in P} (CS_{jpt} + MC_{jpt}).$$

Um diese Werte zusätzlich in Relation zu den vorhandenen Ressourcen zu bringen, bietet sich der relative Ressourcenbedarf RRU_{jt} an:

$$RRU_j = \sum_{p \in P} (CS_{jpt} + MC_{jpt}) / K_p.$$

Mit $\omega_{jt} = RRU_{jt}$ ergibt sich dementsprechend der minimale und maximale relative Ressourcenbedarf MINRRU und MAXRRU.

Bezeichnung	Bedeutung	Modus
MAXSRU	Maximale statischer Ressourcenbedarf	$O = max$
MINSRU	Minimaler statischer Ressourcenbedarf	$O = min$
MAXTRU	Maximaler zeitabhängiger Ressourcenbedarf	$O = max$
MINTRU	Maximaler zeitabhängiger Ressourcenbedarf	$O = min$
MAXRRU	Maximaler relativer Ressourcenbedarf	$O = max$
MINRRU	Minimaler relativer Ressourcenbedarf	$O = min$
MAXDRRU	Maximaler dynamischer relativer Ressourcenbedarf	$O = max$
MINDRRU	Minimaler dynamischer relativer Ressourcenbedarf	$O = min$
MAXTRC	Maximale totale verbliebene Kapazität	$O = max$
MINTRC	Minimale totale verbliebene Kapazität	$O = min$

Tabelle 3.2: Übersicht über alle statischen und dynamischen Prioritätsregeln für das RCPSP/ π .

Dynamische Prioritätsregel Da die in einem Teilplan S tatsächlich zur Verfügung stehende Kapazität einer partiell erneuerbaren Ressource p nicht der Kapazität K_p entspricht, kann man die Regeln MAXRRU und MINRRU durch den dynamischen relativen Ressourcenbedarf $DRRU_{jt}$ verbessern:

$$DRRU_{jt} = \sum_{p \in P} (CS_{jpt} + MC_{jpt}) / K_p^0.$$

Somit entstehen die dynamischen Prioritätsregeln MAXDRRU und MINDRRU. Mithilfe der totalen verbliebenen Kapazität TRU_{jt} , gegeben durch

$$TRC_j = \sum_{p \in P} (K_p^0 - CS_{jpt} - MC_{jpt}),$$

lassen sich ähnliche dynamische Prioritätsregeln MAXTRC und MINTRC erstellen. Für eine Auflistung aller Prioritätsregeln für das RCPSP/ π siehe Tabelle 3.2.

Kapitel 4

Ergebnisse

In diesem Kapitel wird die Wirksamkeit der vorgestellten Algorithmen durch Experimente an generierten RCPSP/ π -Instanzen demonstriert. Alle Daten stammen aus [3].

4.1 ProGen-Erweiterung

Um viele RCPSP/ π -Instanzen generieren zu können, wurde der für RCPSP-Instanzen entworfene Problem-Generator (kurz ProGen) von Kolisch et al. [7] um die zwei Parameter CF und PF ergänzt.

Der Kardinalitätsfaktor $CF \in [0, 1]$ bestimmt die Kardinalität M der Teilmengen Π_p wie folgt,

$$M := \text{ROUND}(1 - CF + T \cdot CF)$$

Für $CF = 0$ gilt, dass die Teilmengen Π_p die minimale Größe einer einzelnen Zeiteinheit abdecken (nur erneuerbare Ressourcen), bei $CF = 1$ decken sie hingegen den gesamten Zeithorizont T ab (nur nicht-erneuerbare Ressourcen).

Der Partitionsfaktor $PF \in [0, 1]$ bestimmt die Anzahl an zusammenhängenden Intervallen I in Π_p :

$$I := \text{ROUND}(1 - PF + \min\{M, T - M\} \cdot CF)$$

Somit werden von einem einzigen zusammenhängenden Intervall, für $PF = 0$, bis hin zu der maximalen Anzahl an unzusammenhängenden Intervallen, bei $PF = 1$, alle möglichen Kombinationen abgedeckt.

4.2 Branch-and-Bound

Für die Evaluierung der Zulässigkeitschranken wurde der Basis-Branch-and-Bound Algorithmus (BV), die mit der ersten Schranke verbesserten Variante (FB 1) und die Variante mit der zweiten Schranke (FB 2) sowie der mit beiden Schranken (FB1&2) erweiterte Algorithmus an

	# of Leaves			CPU-Time in Sec.				# of Leaves			CPU-Time in Sec.		
	AVE	STD	FAC	AVE	STD	FAC		AVE	STD	FAC	AVE	STD	FAC
BV	27,963	188,729	1.0	4.3	26.8	1.0	BV	556,981	863,217	1.0	77.9	119.8	1.0
FB 1	4,441	50,873	6.3	1.0	8.1	4.3	FB 1	123,825	314,838	4.5	20.1	49.6	3.9
FB 2	3,470	35,144	8.1	1.9	16.1	2.3	FB 2	66,360	151,350	8.4	31.2	68.3	2.5
FB 1 & 2	580	6,696	48.1	0.7	5.5	6.0	FB 1 & 2	15,243	39,745	36.5	12.9	33.3	6.1

(a) Durchschnittswerte über alle Instanzen.

(b) Durchschnittswerte über 40 schwere Instanzen.

insgesamt 2.160 generierten Instanzen getestet. Die Instanzen besitzen jeweils 12 Aktivitäten und 30 partiell erneuerbare Ressourcen. In Tabelle 4.1(a) sind die Durchschnittswerte (AVE) und die Standardabweichungen (STD) von der Anzahl an generierten Plänen sowie von der benötigten Berechnungszeit aufgelistet. Der Wert FAC ist der Faktor um den sich die entsprechenden Durchschnittswerte verändern. In Tabelle 4.1(b) sind dieselben Werte nur für die 40 schwersten Instanzen, also die Instanzen bei denen BV die meisten Blätter generiert hat, aufgelistet. Deutlich zusehen ist, dass FB1&2 weitaus weniger Pläne enumeriert als BV bzw. FB 1 und FB 2 alleine. Die Anzahl der Blätter im Präzedenzbaum sind in einem solchen Maße verringert, dass die rechnerisch aufwendigere Variante FB1&2 noch immer schneller ist als beide Schranken einzeln.

Priority Rule ω	217 Instances			14 Instances			250 Instances
	Z = 10	100	1,000	Z = 10	100	1,000	CPU (Z = 1,000)
MINEFT	12.23	9.17	6.38	134.74	133.10	100.40	10.6
MINLFT	1.63	0.81	0.81	7.87	7.87	6.50	8.8
MINSLK	8.70	7.53	6.50	75.13	74.83	40.25	11.3
MTSUCC	4.71	3.31	2.77	55.52	40.06	39.86	11.0
MAXTRC	49.48	9.41	5.26	4.93	3.84	1.63	27.2
MINTRU	49.48	10.01	5.60	4.71	3.95	1.63	19.6
MINRRU	50.15	10.38	5.93	5.04	3.73	1.73	22.3
MINDRRU	50.15	10.38	6.04	5.37	3.73	1.52	32.5

Tabelle 4.1: Vergleich verschiedener Prioritätsregeln an großen Problem-Instanzen. Die prozentuale Abweichung vom optimalen Makespan ist für 217 leichte und 14 schwere Instanzen aufgelistet.

4.3 Prioritätsregeln

Um die Qualität der verschiedenen Prioritätsregeln für GRASP zu vergleichen, wurden 250 große (d.h. mit $J = 32$) Instanzen generiert. Durch das exakte Branch-and-Bound Verfahren wurden für 217 Instanzen optimale Lösungen, für 14 Instanzen zulässige Lösungen und für 19 Instanzen keine Lösungen innerhalb des festgesetzten Zeitlimits von 5 Minuten gefunden. Daher werden die 217 Instanzen als leicht und die 14 Instanzen als schwer angesehen. Die durch GRASP mit verschiedenen Prioritätsregeln gefundenen Lösungen wurden mit den optimalen bzw. zulässigen Plänen des BV verglichen. Die durchschnittlichen prozentualen Abweichungen der Pläne und die

durchschnittlichen CPU-Zeiten von GRASP sind für $Z = 10, 100, 1000$ Durchläufe in Tabelle 4.1 dargestellt. Zu beobachten ist, dass die statischen Prioritätsregeln generell gut bei den einfachen Instanzen abschneiden, bei den schweren jedoch versagen. Dies ist für die dynamischen Regeln genau umgekehrt der Fall. Die Regel *MINLFT* schneidet hingegen im Allgemeinen am Besten ab, da sie auch für schwere Instanzen gute Pläne liefert und dennoch die geringste Rechenzeit verursacht.

Kapitel 5

Zusammenfassung

In dieser Arbeit wurde das Konzept der partiell erneuerbaren Ressourcen eingeführt, und ihre Aussagekraft anhand von logischen Relationen demonstriert. Darüber hinaus wurden zwei Lösungsalgorithmen formuliert und an Beispiel-Instanzen getestet, welche durch ein erweitertes ProGen generiert wurden.

Partiell erneuerbare Ressourcen bringen viele Möglichkeiten durch ihre Modellierungsfähigkeiten mit. Dafür erhöht sich jedoch auch die Komplexität des ohnehin schon schweren RCPSPs. Optimale Verfahren, wie das vorgestellte exakte Branch-and-Bound Verfahren, sind daher für reale Problemstellungen mit einer hohen Zahl an Bedingungen nicht sinnvoll einsetzbar. Allerdings lassen sich durch heuristische Verfahren wie GRASP auch die erhöhten Anforderungen des RCPSP/ π einigermaßen bewältigen.

Partiell erneuerbare Ressourcen sind daher eine sinnvolle Erweiterung des RCPSP, wenn es die Anwendung erfordert.

Literaturverzeichnis

- [1] A. Schirmer and A. Drexl. Allocation of partially renewable resources: Concept, capabilities, and applications. *Networks*, 37:21–34, 2001.
- [2] E. Demeulemeester and W. Herroelen. A branch-and-bound procedure for the multiple resource-constrained project scheduling problem. *Management Science*, 38(12):1803–1818, 1992.
- [3] J. Böttcher, A. Drexl, R. Kolisch, and F. Salewski. Project scheduling under partially renewable resource constraints. *Management Science*, 45:544–559, 1999.
- [4] J. H. Patterson, B. Talbot, R. Slowinski, and J. Weglarz. Computational experience with a backtracking algorithm for solving a general class of precedence and resource-constrained scheduling problems. *European Journal of Operational Research*, 49:68–79, 1990.
- [5] P. Brucker and S. Knust. *Complex Scheduling*. Springer, 2006.
- [6] R. Kolisch and A. Drexl. Adaptive search for solving hard project scheduling problems. *Naval Research Logistics*, 43:23–40, 1998.
- [7] R. Kolisch, A. Sprecher, and A. Drexl. Characterization and generation of a general class of resource-constrained project scheduling problems. *Management Science*, 41(10):1693–1703, oct 1995.