数理解析研究所講究録
1194巻 2001年 97-104

ソフトウェア開発プロジェクトの最適な計画立案方法

関西大学大学院総合情報学研究科博士課程　伊佐田百合子(Yuriko Isada)
Faculty of Informatics, Kansai University Gradate School
関西大学総合情報学部　仲川勇二 (Yuji Nakagawa)
Faculty of Informatics, Kansai University

1. はじめに

「情報技術は重要な経営資源である」という認識が浸透するにしたがって，情報技術を事業戦略に活用する企業が急増しており，インターネットを利用した新しいビジネスが急速に展開されている．コンピュータシステムが，作業効率化のためのツールから企業戦略そのものに変貌する中で，システムの大規模化，複雑化，多様化が進んでいる．現在，企業を取り巻く経営環境は，「ドッグイヤー」，「マウスイヤー」といわれるように，非常に早いスピードで動いている．このような環境下で，ソフトウェア開発プロジェクトは，３ヶ月，遅くとも６ヶ月以内にシステム構築を完了し，経営効果を出さなければならない．したがって，ソフトウェアに対するさまざまな要求を満たしつつ，納期までに開発を完了することは非常に重要なことである．ソフトウェア開発プロジェクトを成功させるためには，適切な開発計画を立案し，進捗状況により的確にスケジュールの見直しと対策が実施できることが重要である．しかし，ソフトウェア開発プロジェクトを開始する段階で，規模を正確に見積もること，及び，リスクを予想することは非常に困難である．そのため，適切な開発計画を立案することは非常に難しい．また，ソフトウェア開発プロジェクトの目的は，単独であることは少なく，複数の目的が存在することが一般的である．それらの目的は競合することが多く，すべての目的を同時に最大化（または，最小化）するような解は存在しない．したがって，複数の開発目的を総合的に評価して満足できるソフトウェア開発計画を立案する必要がある．すなわち，ソフトウェア開発計画問題は，多目的最適化問題である．さらに，複数のリソースで複数の作業工程を実行することから，組合せ最適化問題の１つであるスケジューリング問題ともいえる．

古宮ら[1]は，ソフトウェア開発における作業工程間の先行関係，及び，リソースの割当て条件などの制約事項により解候補となる組合せ数を吟味することで組合せの数の増加を押さえ，遺伝的アルゴリズムを適用したソフトウェア

開発計画の立案方法を提案した．本稿では，ソフトウェア開発計画問題を多目的離散最適化問題と位置づけ，多目的離散最適化問題の解法を用いた意思決定支援のアルゴリズム[2] [3]がソフトウェア開発計画立案に有効であることを示す．

2. ソフトウェア開発計画問題

2.1. ソフトウェア開発計画の特徴

最適な計画を策定するために，考慮すべきソフトウェア開発計画の２つの特徴について明らかにする．

はじめに，ソフトウェア開発管理手法の違いに見られるソフトウェア開発計画の特徴である．ソフトウェア開発工程の管理にはさまざまなモデルが適用される．代表的なモデルとして，ウォータフォール型モデルとプロトタイピング型モデルをあげることができる．ウォータフォール型モデルは，ソフトウェア開発を職人芸的な作成法から，近代的な工業製品としての製作法に変えることが検討された結果，考え出されたソフトウェア開発方法論[4]であり，広く世界中で活用されている伝統的なソフトウェア開発工程管理モデルである．しかし，現実のソフトウェア開発では，要求仕様の変更などにより，工程のやり直しが発生することが一般的である．工程のやり直しによる無駄な作業を避けるために考え出された方法が，プロトタイピング型モデルである．プロトタイピング型モデルは，ソフトウェアの主要機能を開発して，ユーザに公開し，ソフトウェアの完成度を高めていく方法である[5]．システムを取り巻く環境の変化が激しく，要求仕様を確定してソフトウェア開発を開始することが困難な場合に適したソフトウェア開発工程管理モデルである．ソフトウェア開発工程管理モデルの上位モデルと位置づけられるモデルに，SDEM90[6]がある．これは，ソフトウェア開発の構造をメタモデルとして表現したもので，システム開発における役割や仕事を網羅的に表現している．システムが動作する環境を 23 にカテゴリ化し，各カテゴリに対して 11 の作業レベルを設定する．カテゴリと作業レベルのマトッリックスをとると，253 の交点ができあがる．この 253 個の交点の中から意味ある交点として153の交点を選びだし定義したものがSDEM90である．SDEM90 は，ソフトウェア開発の構造モデルであり，どの工程をどのような手順で組み合わせるのかは自由であるため，開発手順が異なる開発モデルであっても，また，開発プラットホーム，及び，開発メソッドが変化しても汎用的に使用することができる．したがって，このモデルをベースにしたソフトウェア開発計画の立案方法は，汎用的に活用することが可能である．

次に，ソフトウェア開発の複雑性にみられるソフトウェア開発の特徴である．ビジネスソフトウェアの開発は，社会，経済の法則，すなわち，人間の活動そのものをシステムで記述しようとする試みである．これらの仕組みが複雑化するにしたがって，ソフトウェア開発の複雑性は増大する．自然の物理法則にしたがって作成される人工構造物とは全く異なる複雑性と規模を持っている[5] [7]．これは，工業製品になぞらえたソフトウェア開発工程の管理手法であるウォータフォール型モデルにおいて，工業製品と同等の管理を実現できなかった理由でもある．現在，ソフトウェア開発を支援するツールが提供され，部分的には自動化が進められてきている．しかし，ソフトウェア開発の大部分は，ソフトウェア開発プロジェクトにおける集団的知的生産活動であり，さまざまな技術，及び，スキルレベルが要求される．ソフトウェアの構造物の特性である複雑性と，ソフトウェア開発プロジェクトの組織としての複雑性が相乗効果となり，ソフトウェア開発の複雑性を高めている．

2.2. ソフトウェア開発計画問題

ソフトウェア開発において，m個の作業工程がn個のリソースによって処理されるとする．作業工程間には，先行関係が存在し，ある作業工程は，特定のリソースによってのみ実施可能とするとき，以下の目的を満足するように，全作業工程に最適なリソースを割当て，ソフトウェア開発計画を立案することを検討する．

(1) ソフトウェアの信頼性の評価値を最大にする
(2) ソフトウェアの保守性の評価値を最大にする
(3) ソフトウェアの開発期間を最小にする
(4) 総コストを最小にする

ソフトウェア開発計画立案時点における信頼性は，予想残存バグ数を尺度として測定することができる．残存バグ数は，過去の開発実績から，類似の規模，機能数をもつソフトウェア開発の計測結果を用いて推定する．作業工程iで混入した残存バグ数をR_iとしたとき，それぞれの作業工程がすべて同等の重要度をもつわけではないので，重要度，発生個所による重みづけω_iを考慮し，残存バグ数の平均値の最小化を図る．ここで，R_iは，離散的な値をとるものとする．

保守性は，以下の点について実施目標を評点化し，評価する．

(1) 運用機能の作りこみ度合い
(2) コメントの記述量

(3)　構造化のレベル

(4)　部品の再利用度

保守性を向上させるための要素がｎ個あるとすると，作業工程ｉの保守性M_iは，次のように表される．

$$M_i = \sum_{j=1}^{n} o_{ij} = o_{i1} + o_{i2} + \cdots + o_{in} \qquad i = (1,2,\cdots m).$$

ここで，M_i，及び，o_{ij}は，離散的な値をとるものとする．

ソフトウェア開発における各作業工程間には，通常，先行関係が存在する．ソフトウェア開発計画をアローダイアグラムで図示した時，ソフトウェア開発期間は，アローダイアグラム上の始点から終点までの最大長路（クリティカル・パス）で示すことができる．開発期間を最小化することは，クリティカル・パス上の余裕時間（フロート）を最小化することに等しい．l個の経路があり，tを経路別の開発時間とする時，ソフトウェアの開発期間$T(\mathbf{x})$は，次のように表される．

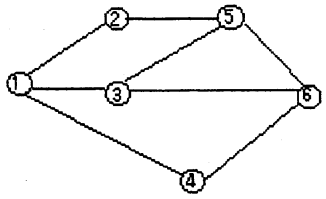

図１　経路

$$T(\mathbf{x}) = \max\{t_1,\cdots,t_l\}.$$

例として，６つの作業工程からなるソフトウェア開発プロジェクトのアローダイアグラムを図１に示す．この時の開発期間$T(\mathbf{x})$は，以下の式であらわすことができる．

$$T(x) = \max\{t_1(x_1,x_2,x_5,x_6), t_2(x_1,x_3,x_5,x_6), t_3(x_1,x_2,x_5,x_3,x_6), t_4(x_1,x_3,x_6), t_5(x_1,x_4,x_6)\}.$$

ソフトウェアの開発コストは，どの作業工程にどの要員を割り当てるかということにくわえて，信頼性を高めるために要する費用，保守性を向上させるために要する費用が考えられる．ｎ個の費用要因があるとすると，作業工程ｉの開発コストC_iは，次のように表される．

$$C_i = \sum_{j=1}^{n} c_{ij} = c_{i1} + c_{i2} + \cdots + c_{in} \qquad i = (1,2,\cdots m).$$

ここで，C_i，及び，c_{ij}は，離散的な値（整数値）をとるものとする．

作業工程ｉの予想残存バグ数を$R_i(x_i)$，保守性を$M_i(x_i)$，開発期間を$T(\mathbf{x})$，開発コストを$C_i(x_i)$とおく．ここで，変数x_iはその項目案であり，作業工程ｉにおける項目案番号である．予定された費用をｂとしたとき，ソフトウェア開発計画問題は，以下のように定式化することができる．

$$\text{Min} \quad f_1(\mathbf{x}) = \frac{1}{m}\sum_{i=1}^{m} \varpi_i R_i(x_i)$$

$$f_2(\mathbf{x}) = \sum_{i=1}^{m} -M(x_i)$$

$$f_3(\mathbf{x}) = T(\mathbf{x})$$

$$\text{s.t.} \quad g(\mathbf{x}) = \sum_{i=1}^{m} C_i(x_i) \le b$$

$$x_i \in \{1,2,\cdots,K_i\}\,(i = 1,2,\cdots,m).$$

ここで． $\mathbf{x} = \{x_1,\cdots,x_m\}$, $f_i(k) = a_{ik}\ (k = 1,2\cdots,K_i, i = 1,2,\cdots,m)$である．

3. 多目的離散最適化法の適用

ソフトウェア開発プロジェクトにおいて，最適な開発計画を立案するために，ソフトウェア開発計画問題に多目的離散最適化法を適用する．多目的離散最適化法は，多目的離散最適化問題の複数の目的関数を単一の目的関数（代理目的関数）に変換し，仲川[8]により提案されたモジュラ法を適用して，実行可能解を列挙する方法である．多目的離散最適化問題の複数の目的関数を単一の目的関数に変換する問題を，代理問題と呼び，標的値以上の目的関数の値を持つ実行可能解を列挙する問題を，単一標的問題と呼ぶ．単一標的問題を解いて得られた解は，標的解と呼ばれる．単一標的問題は，次のように定式化される．

$$\textbf{target} \quad \mathbf{uf}(\mathbf{x}) \ge f^{ST}$$

$$\textbf{s.t.} \quad g(\mathbf{x}) \le b$$

$$\text{但し、} \sum_{i=1}^{l} u_i = 1, u_i \ge 0.$$

ここで，$\mathbf{f}(\mathbf{x}) = \{f_1(\mathbf{x}),\cdots,f_l(\mathbf{x})\}$は，$l$次元目的関数ベクトル，$\mathbf{u} = \{u_1,\cdots,u_l\}$は，代理乗数ベクトル，$\mathbf{uf}(\mathbf{x})$は代理目的関数，$f^{ST}$は代理目的の標的値である．代理乗数を変化させることにより，目的関数の優先度を調整することが可能である．

モジュラ法を適用した多目的離散最適化問題の解法の計算例を表 1に示す．テスト問題は，擬似乱数$1 \le f_{ij}(k+1) - f_{ij}(k) \le 2^8$を使用して生成し，3 目的，10 代替案で，変数の数を 100，200, 400, 600, 800, 1000 と変化させ，それぞれのケースで，解の数と処理時間を測定した．計算実験には，ＣＰＵ500MHz，メモリー192MB の DOS/Ｖコンピュータを使用した．計算結果は，現在まで解くことが

表 1　テスト問題 1（3 目的 10 代替案 100, 200, 400, 800, 1000 変数）

	解の数	処理時間（秒）	最大代替案数		解の数	処理時間（秒）	最大代替案数
3x100x10	3	0	669	3x600x10	3	9	627
	10	0	2286		10	9	1014
	20	0	4088		20	8	1498
	30	1	5248		30	9	2320
	50	0	7500		50	10	4580
3x200x10	3	0	224	3x800x10	3	16	1630
	10	1	588		10	17	5894
	20	1	1409		20	17	12112
	30	1	2298		30	20	21987
	50	1	3033		50	19	23577
3x400x10	3	3	2116	3x1000x10	3	28	1039
	10	3	4763		10	28	1232
	20	3	9420		20	29	2759
	30	4	14464		30	30	4833
	50	5	24812		50	31	9035

困難であった大規模な問題を一般的に使用されるコンピュータにおいて実用的な時間内で解きうることを示している．仲川[9]らは，モジュラ法を非線型ナップザック問題に適用し１０００の変数を持つ大規模な問題を解きうることを示した．

さらに，実際のソフトウェア開発計画問題に適用することが可能であること検証するため，サブシステム数が 5，8，12 で，それぞれ対応する開発要員数が 5，25，50 の開発プロジェクト想定する．ソフトウェア開発工程は，サブシステム毎に 153 の工程に分割することが可能であるから，目的関数の数を 3 とした場合，次のような規模の問題を解くことが必要となる．

- 3 目的，765 変数，5 項目
- 3 目的，1224 変数，25 項目
- 3 目的，1836 変数，50 項目

表 2　テスト問題 2

	解の数	処理時間（秒）	最大代替案数
3x765x5	3	9	1214
	6	10	1310
3x1224x25	4	80	1522
	6	80	1696
3x1836x50	4	329	4615
	6	329	6091

擬似的なソフトウェア開発計画問題への適応結果を表 2に示す．ここでもテスト問題は，擬似乱数 $1 \le f_{ij}(k+1) - f_{ij}(k) \le 2^8$ を使用して生成し，ＣＰＵ500MHz，メモリー192MB の DOS/Ｖコンピュータを使用した．この結果，多目的離散最適化法を用いることにより，一般的なパーソナルコンピ

ュータでも，実際のソフトウェア計画問題を実用的な時間内に解きうることが明らかである．

4. 開発計画立案方法

多目的離散最適化法を適用して，ソフトウェア開発プロジェクトの計画を立案する場合，開発期間は，変数分離ではないため，開発期間を除く目的関数に対して，モジュラ法を用いて解き，実行可能解を列挙する必要がある．得られた実行可能解について，開発期間をもとめ，すべての目的関数に関して優越性テストを実施すれば，パレート最適解を得ることができる．また，多目的最適化法は，最大化問題を解くように設計されているため，信頼性の評価値である残存バグ数の最小化は，テスト時のバグ削減率の最大化に置き換えて解く必要がある．

以下に，多目的離散最適化法を用いたソフトウェア開発プロジェクトの開発計画を立案するための手順を述べる．

（ステップ１）開発計画を立案するソフトウェア開発において，各リソースを作業工程に割り当てた場合の，コストと，作業期間を設定する．さらに，ソフトウェア開発の目的についてソフトウェアの評価指標を設定し，各作業工程の評価値を設定する．

（ステップ2）代理乗数uの初期値を設定し，代理問題をモジュラ法を用いて解き，代理目的関数の目的関数の値をもとめ，初期標的値 f^{ST} を定める．

（ステップ3）単一標的問題を多目的離散最適化法を用いて，標的解を列挙する．列挙された標的解の数が適切であれば，次のステップに進み，そうでない場合は，標的値 f^{ST} を更新して，このステップを再度実行する．

（ステップ４）得られた標的解に対して，経路別に開発期間をもとめる．経路別の開発期間の最大値が，ソフトウェア開発プロジェクトの開発期間であるので，開発期間の目的関数値を経路別開発期間の最大値に定める

（ステップ５）各目的関数値に対して優越性操作を行い，パレート最適解をもとめる．

（ステップ６）現在のパレート解集合の中で，意思決定者の価値観にあう解が見つかれば，ここで探索を終了し，見つからなければ，代理乗数を更新し，ステップ２に戻る．

5. おわりに

本論文では，多目的離散最適化問題を適用した意思決定アルゴリズムを用いて，ソフトウェア開発計画を立案することを提案した．ソフトウェア開発プロジェクトの途中で，トレッキングした信頼性，保守性の各評価指標の実測値をこのアルゴリズムに組み込むことによって，ソフトウェア開発の実態に即応したスケジュールの見直しを行うことが可能となり，最適なプロジェクト推進が可能になると予測される．また，多目的離散最適化問題の解法を用いて，一般的に使用されているパソコンレベルで大規模な問題を解きうることから，作業工程数が増加し，問題の規模が大きくなっても，計画立案が可能であると予想される．

今後の研究の課題として，最適な代理乗数，及び，標的解のサイズ設定の研究と，ユーザインターフェイスの改善があげられる．

参考文献

[1] 古宮，澤部，櫨山：「制約に基づくソフトウェア開発計画の立案」，電子情報通信学会論文誌，Vol.J79-D-I，No.9，pp.554-557（1996）．
[2] Y.Isada, M.Hikita, Y.Nakagawa : A Method for Solving Multi-objective Discrete Optimization Problem, Proceedings of the first western pacific and third australia-japan workshop on stochastic models in engineering, technology and management, pp.192-201 (1999).
[3] 疋田，仲川：「多目的離散最適化問題のための対話型意思決定アルゴリズム」，経営工学会論文誌，Vol.51，pp.197-202（2000）．
[4] 菅野：「ソフトウェア開発のマネージメント」，（株）新紀元社（1984）
[5] L.H.Putnam，W.Myers，研野和人訳：「プロジェクトの見積りと管理のポイント」，共立出版（株）（1995）
[6] 板倉：「スーパーSE」，日科技連出版社（1993）
[7] Frederick P. Brooks, Jr. :「No Silver Bullet: Essence and Accidents of Software Engineering」, Computer, pp.10-19 (1987)
[8] 仲川：「離散最適化のための新解法」，電子情報通信学会論文誌，Vol.J73-A，No.3，pp.550-556（1990）．
[9] Y. Nakagawa, A. Iwasaki : Modular Approach for Solving Nonlinear Knapsack Problems, IEICE TRANS.FUNDAMENTALS, VOL.E82-A, NO.9, pp1860-1864 (1999).