

# コンピュータ分野における TRIZ矛盾表の適用

## —アーキテクチャ・方式分野—

2007/8/30

庄内 亨 (日立製作所 中央研究所)  
河辺 峻 (明星大学 情報学部)\*  
濱中 直樹 (日立製作所 マイクロデバイス事業部)\*

\*: 本研究は日立製作所中央研究所に在籍時のものです。

© Hitachi, Ltd. 2007. All rights reserved.

## コンピュータ分野における TRIZ矛盾表の適用

### Contents

1. はじめに
  - ・問題点と対応方針
  - ・関連研究との関係
2. 言葉の変換、サブマトリクス化
  - ・事例: プロセッサ、キャッシュ(2件)、位相調整
  - ・変換(言換え)のまとめ、サブマトリクス
3. 適用範囲の拡大
  - ・並列機、ストレージ、DB、システム管理等への拡大
4. 特許創生への適用事例
  - ・データセンタアーキテクチャ(ブレードシステム)
5. おわりに(結論など)

- コンピュータやソフト関係者がよくする質問
  - 「TRIZは自分の分野でも使える？」
- 原因は
  - 「矛盾表」と「発明原理」が、「機械」・「構造」・「物理」・「化学」に偏ってみえる。
- 対策
  - 事例を集める、
  - コンピュータ分野で連想に使える言葉を増やす
  - 関係の薄い改善・悪化パラメータを隠す

## 1.2 関連研究

- TRIZ and Software — 40 Principle Analogies[Rea]
  - 分野が我々と異なり、適用が難しい。
- NECにおけるTRIZ推進活動[鈴木他]
  - 組織的取組みを説明。
- Re-structuring TRIZ to Software Engineering[Mann]
- ソフトウェア工学とTRIZ[中川]
  - ソフトウェア開発プロセスが対象領域であり、分野が異なる。
- Matrix 2003[Mann, et al.]
  - 改善/悪化パラメータはコンピュータ分野を考慮済み。
  - 発明原理は従来のまま。

★「コンピュータ」や「ソフトウェア」と言っても、分野が広大。  
★自分らの領域に合致しない可能性も。

- 内容
  - マルチバンクのキャッシュにおいてロード命令の組が起こすバンク競合を回避したい。
- TRIZ適用
  - 悪化パラメータ → 6 面積が増える(マルチポートキャッシュによるチップ面積の増加)
  - 改善パラメータ → 25 時間の損失(性能を上げたい)
  - 発明原理 → 10 アクションの先取り、35状態の変移、17 他次元への転換、4 非対称性
- アイデアの核心
  - 予測テーブルで履歴を作成し、競合を起こさないロード命令の組を予測し、発行する。
- TRIZ 的分析
  - 35 状態の変移を予測することにより 10 アクションの先取りを行う。
- 解釈・コメント
  - 「状態の変移を予測することによりアクションの先取りを行う」という形の発想は分岐予測やメモリデータ予測などを初めとするコンピュータの高速処理における基本的考えである。

- 内容
  - 対称型マルチプロセッサ(SMP)ではプロセッサ数の増加に伴い、Storeのスループット低減が課題で、Store-in-cacheが必須である。実装制約により、すべてのプロセッサでキャッシュを完全共有するのは困難で複数のStore-in-cacheが必要となる。複数のStore-in-cache間の一致制御のオーバヘッドを減らしたい。
- TRIZ適用
  - 悪化パラメータ → 24 情報が失われる(一致制御の失敗)
  - 改善パラメータ → 25 時間の損失(性能を上げたい)
  - 発明原理 → 24 仲介、26 コピー化、28 方式の転換、32 色を変える
- アイデアの核心
  - データの参照形態を参照履歴を用いて予測し、その結果を活用して、動的にキャッシュ一致制御を最適化する。
- TRIZ 的分析
  - データに 32 色をつけ(データの参照形態を参照履歴を用いて予測し)、そのデータに関しては 26コピー化して共有化する。
- 解釈・コメント
  - 32 色を変えるという発明原理がたびたび登場する。コンピュータのアーキテクチャ・方式・論理にとって、これは「情報にタグをつけて他と区別する」という意味に解釈すべきで、非常に有効な発明原理である。

- 内容
  - キャッシュメモリの容量には上限があり、限られた容量の中で、如何にしてヒット率を向上させるかが課題。局所性の高いデータを見つけ、区別してヒット率を向上させたい。
- TRIZ適用
  - 悪化パラメータ → 24 情報が失われる(一致制御の失敗)
  - 改善パラメータ → 25 時間の損失(性能を上げたい)
  - 発明原理 → 24 仲介、26 コピー化、28 方式の転換、32 色を変える
- アイデアの核心
  - ある種のデータをその他のデータより優先的に格納でき、どれを優先するかを決めるポリシーをシステムやパーティション毎にプログラムの性質を考慮して変更する
- TRIZ 的分析
  - データに 32 色をつけ(局所性の高いデータにタグをつけて他と区別する)、24 仲介ポイント(切り替えポイント:パーティション毎、システム構成時、ブート時、...)毎に変更可能にする。
- 解釈・コメント
  - 24仲介 という発明原理もたびたび登場する。コンピュータのアーキテクチャ・方式・論理にとってみると、ブロック転送発生時とか、割り込み発生時とか、レベルによりいろいろ考えられる。何かの切り替えポイントと考え、そのポイントを活用する特許アイデアがある。

- 内容
  - LSI チップ間の信号インタフェース回路、特に装置間インタフェースのようにデータビット間での遅延時間ばらつきが大きいインタフェース回路において、スキューばらつきを改善したい。
- TRIZ適用
  - 悪化パラメータ → 9 チップ面積が増える
  - 改善パラメータ → 31 悪い副作用(スキューばらつきを改善したい)
  - 発明原理 → 22 害を益に変換、1 細分化、40 複合材料
- アイデアの核心
  - 送信側から伝送されるデータ信号と並送クロック信号との位相差を複数点で検出して保持し、その情報を基に送信側からのデータ信号の位相を調整する。
- TRIZ 的分析
  - 22 害(位相差大)をみて益(位相調整)に変換。1 細分化(部分化により詳細に制御)。
- 解釈・コメント
  - この矛盾マトリクスにたどり着くのに少し時間がかかった。そもそも 31 悪い副作用とは抽象的で良く判らないが、この場合は「スキューばらつきが拡大する」がこれに相当すると考える。提示される発明原理は 22 害を益に変換 であるが、この特許アイデアを見るとピッタリである。しかし、「害を益に変換」という言葉だけでこのアイデアが生まれるためには、相当の技術的専門知識が必要である。

## 2.2 具体例の分析のまとめ

- 発明原理の言換え
  - 32 色を変える → 情報にタグをつけて他と区別する
  - 24 仲介 → ブロック転送発生時とか、割り込み発生時とか、レベルによりいろいろ考えられる。何かの切り替えポイントと考え、そのポイントを活用して何かを起こす。
- 改良/悪化パラメータの言い換え
  - 性能を上げたい → 25 時間の損失、
  - 論理規模の増大、チップ面積の増大 → 6 静止物体の面積
  - キャッシュー貫性保障 → 24 情報が失われる
- サブマトリクス化
  - アーキテクチャ方式にとって関係の薄いパラメータを思い切って削除し、14のパラメータのみを残す。
  - マトリクスの面積が1/4 以下に。
  - A4一枚やPC の800x600 の画面に収めることができ、初心者でもTRIZの適用がやり易くなる。

## 2.2の続き サブマトリクス

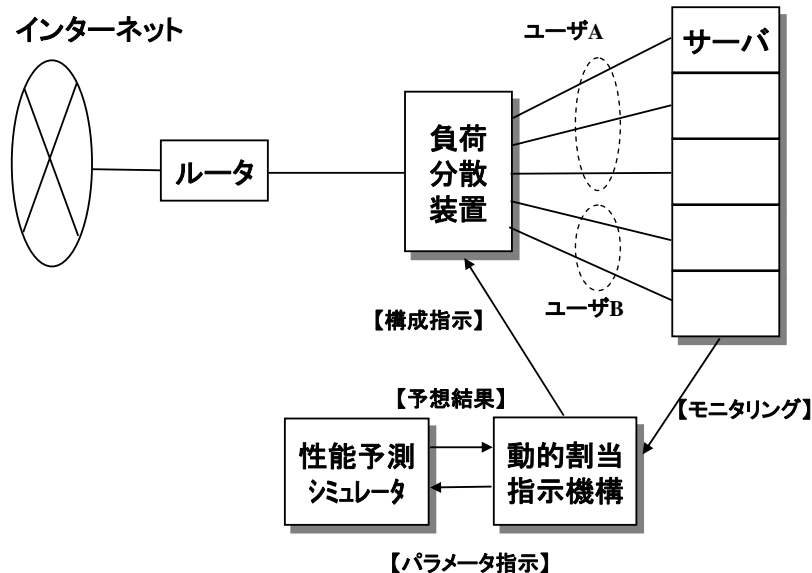
悪化パラメータ→ 改善パラメータ ↓	6	9	10	12	15	16	24	25	26	27	30	31	33	34
	不 動 物 体 の 面 積	速 度	力	形 状	動 く 物 体 の 運 動 の 持 続 性	不 動 物 体 の 運 動 の 持 続 性	情 報 の 損 失	時 間 の 損 失	物 質 の 量	信 頼 性	物 体 に 働 く 有 害 要 因	悪 い 副 作 用	操 作 の 容 易 さ	保 守 の 容 易 さ
6 不動物体の面積	+	-	1,18,35 .36	-	-	2,10,19 .30	30,16	10,35,4 .18	2,18,40 .4	32,35,4 .35	27,2,39 .35	22,1,40	16,4	16
9 速度	-	+	13,28,1 5,19	35,15,1 8,34	3,19,35 .5	-	13,26	-	10,19,2 9,38	11,35,2 7,28	1,28,35 .23	2,24,35 .21	32,28,1 3,12	34,2,28 .27
10 力	1,18,36 .37	13,28,3 5,12	+	10,35,4 0,34	19,2	-	-	10,37,3 6	14,29,1 8,36	3,35,13 .21	1,35,40 .18	13,3,36 .24	1,28,3 .25	15,1,11
12 形状	-	35,15,3 4,18	35,10,3 7,40	+	14,26,9 .25	-	-	14,10,3 4,17	36,22	10,40,1 6	22,1,2 .35	35,1	32,15,2 6	2,13,1
15 動く物体の運動の持続性	-	3,35,5	19,2,16	+	14,26,2 8,25	-	10	20,10,2 8,18	3,35,10 .40	11,2,13	22,15,3 3,28	21,39,1 6,22	12,27	29,10,2 7
16 不動物体の運動の持続性	-	-	-	-	-	+	10	28,20,1 0,16	3,35,31	34,27,6 .40	17,1,40 .33	22	1	1
24 情報の損失	30,16	26,32	-	-	10	10	+	24,26,2 8,32	24,28,3 5	10,28,2 3	22,10,1	10,21,2 2	27,22	-
25 時間の損失	10,35,1 7,4	-	10,37,3 6,5	4,10,34 .17	20,10,2 8,18	28,20,1 0,16	24,26,2 8,32	+	35,38,1 8,16	10,30,4	35,18,3 4	35,22,1 8,39	4,28,10 .34	32,1,10
26 物質の量	2,18,40 .4	35,29,3 4,28	35,14,3	35,14	3,35,10 .40	3,35,31	24,28,3 5	35,38,1 8,16	+	18,3,28 40	35,33,2 9,31	3,35,40 .39	35,29,2 5,10	2,32,10 .25
27 信頼性	32,35,4 0,4	21,35,1 1,28	8,28,10 .3	35,1,16 .11	2,35,3 .25	34,27,6 .40	10,28	10,30,4	21,28,4 0,3	+	27,35,2 40	35,2,40 .26	27,17,4 0	1,11
30 物体に働く有害要因	27,2,39 .35	21,22,3 5,28	13,35,3 9,18	22,1,3 35	22,15,3 3,28	17,1,40 .33	22,10,2	35,18,3 4	35,33,2 9,31	27,24,2 40	+	-	2,25,28 .39	35,10,2
31 悪い副作用	22,1,40	35,28,3 .23	35,28,1 40	35,1	15,22,3 3,31	21,39,1 6,22	10,21,2 9	1,22	3,24,39 .1	24,2,40 .39	-	+	-	-
33 操作の容易さ	18,16,1 5,39	18,13,3 4	28,13,3 5	15,34,2 9,28	29,3,8 25	1,16,25	4,10,27 .22	4,28,10 .34	12,35	17,27,8 40	2,25,28 .39	-	+	12,26,1 .32
34 保守の容易さ	16,25	34,9	1,11,10	1,13,2 4	11,29,2 8,27	1	-	32,1,10 .25	2,28,10 .25	11,10,1 .16	35,10,2 .16	-	1,12,26 .15	+

- 拡大の範囲
  - 並列計算機、ストレージ装置(RAID、NAS)、データベースシステム、ITシステム管理、画像・音声・言語処理など
- 言い換えや発明原理への事例の追加
- サブマトリクスの修正
  - 32 製造の容易さ, 36 装置の複雑さ, 37 検出と測定の困難さ, 39 生産性 の追加、の4パラメータの追加を行った。
- 「発明原理の言い換え」の意義
  - 各分野の技術的知見をエッセンス化
  - あるレベルの研究者には、分野/部門での技術的問題解決で常套的に活用される解決戦略が暗黙知として蓄積されている。
  - この暗黙知を明文化でき、分野に精通していない人に、その分野の主要な解決戦略を伝授できる。

- インターネット・データセンタ(iDC)での問題
- アクセス集中による応答性能の悪化等が問題化
  - i-modeがシステム障害となり、サービス停止に発展し、社会的な問題に(1999年～2000年頃)
- 多くの検討から、特許案を考案した。
  - 「仮想プライベート・データセンタ」Virtual Private Data Center (VPDC)
  - データセンタ内にある機器(サーバやストレージなど)を複数のユーザや業務で共有して利用効率を向上させると共に、予想を超えるインターネットからのアクセス集中への対応を可能にするデータセンタ・アーキテクチャ
- コンセプトの基本特許化の過程で、TRIZ 適用

### 問題の定式化

- ① インターネットからの急激な負荷に対応してサーバ構成を変化させ、最適化によりスループットを上げる。
- ② サーバ数(全体の物量)を減らす。(コスト削減のため)
- ③ 最適化を操作性良く行う。



© Hitachi, Ltd. 2007. All rights reserved. 12

## 4. 特許創生への適用事例(3/4)

### 矛盾問題(1)

- 改善:サーバ構成を最適化してスループットを上げたい……①
- 悪化:サーバ数(物量)も減らしたい。……………②

### 発明原理

- 35 状態の変移、38 強力酸化剤を使う、18 振動・発信・高周波、16 部分/過剰解決

### 解釈

- 状態の変移をとらえて(予測して)部分的解決を図るか、または過剰解決を図る、その際に周期的・波動的な動きを考慮する
- 「状態の変移」は、サーバのCPU効率の変移、サーバの主記憶効率の変移、サーバのI/O 効率の変移などであり、これらがモニタリングされることが望ましい。
- 「部分的解決または過剰解決」は、限られたリソース(サーバ数)の中での最適配置とも考えられる。
- 「周期的・波動的な動き」は、負荷の周期的変動とも考えられる。

### 最適配置と③ → 矛盾問題

© Hitachi, Ltd. 2007. All rights reserved. 13

- 矛盾問題(2)
  - 改善:最適配置を簡単に行う。操作性良くしたい。……③
  - 悪化:操作性良くしようとすると時間がかかる。
- 発明原理
  - 4 非対称性、28 方式の転換、10 アクションの先取り、34 部分の放棄・変形、又は再生成。
- 解釈
  - モデル化によるシミュレーションや実測を行い、先を予測することにより、先行して最適配置をする(アクションの先取り)
  - サーバ等の放棄・再利用により最適配置をする(部分の放棄・変形または再生成)
- アイデアの一部は日立のサービスプラットフォームコンセプトHarmonious Computingや統合サービスプラットフォーム BladeSymphonyに活かされた

- 結論
  - 矛盾マトリクスや発明原理をコンピュータ分野の特許創生へ適用した際に必要であった発明原理の言い換えやサブマトリクス化について述べた。
  - これらは、TRIZを組織的に適用する際に有効である。
- しかし
  - これだけで十分な効果が得られる訳でもない。
  - 技術課題やアイデアの表層のみの理解に基づくTRIZの適用では効果はない。
  - 技術や問題の全体を深く把握・理解し、状況に応じて最適なTRIZツールをうまく適用し、従来の思考方法も捨てずに活用することも重要である。
- Matrix 2003について
  - 改良/悪化パラメータはコンピュータ分野も考慮済み
  - 発明原理は以前のもままであり、本アプローチはMatrix 2003にも有効と思われる。