6J-4 容量を考慮したネットワークの信頼度

柳　　繁　（防衛大学校）
浅上　勉　（防衛大学校）
佐々木正文　（防衛大学校）

1．序論

本報告では，ネットワークの特定の2つのノード間の最大フロウが規定値以上である確率をネットワークの信頼度と定義し，この信頼度を効率的に求める方法を提案する．この問題に対する従来の研究としては，1）全ての最小カット集合を求めて計算する方法［1，2］，2）陰的列挙法による方法［3］の2つがある．本報告は後者の部類に属する．後者の方法は以下の手順よりなる．

i)ネットワークの各要素に番号を付与する．

ii)ネットワークの要素からなる部分集合に対して，要素数の少ない順に，また，同じ要素数の場合は要素番号についての辞書的順序によって部分集合を列挙する順序を決定する．

iii)順序に従って陰的列挙を行う．この時，ネットワークのリンクまたはカットが見つかる度に陰的列挙の為の条件を更新・削除する．

この方法は前者に比べると効率はよいが列挙の順序が全てのネットワークに対して一様に決められているために，陰的列挙が効果的に適用されていない．本報告では，この点を改良し，陰的列挙条件に関する分岐限定法を適用することによりアルゴリズムの改良を行う．

2．仮定

1）ネットワークの要素($a_1, a_2, ..., a_n$)は枝とノードであり，これらの要素の状態は正常，故障の2状態をとる．

2）各要素には容量制限がある．

3）枝は，有向でも無向でも良い．

4）各要素は独立である．

5）ネットワークは単調構造である（コヒーレントシステム）．

3．　記号

n：ネットワークの要素数

e_i：a_iが正常である事象

$\bar{e}_i$：a_iが故障である事象

c_i：a_iの容量制限

p_i：a_iの信頼度

$\bar{p}_i$：a_iの故障確率

P_N：ネットワークの信頼度

$mf(a_i, a_j)$：a_i，a_j間の最大フロウ

Pr｛　｝：確率

リンク：規定の2要素（通常，ノード）間のフロウを規定値以上とする要素の集合

push：スタックに書き込み

pop：　スタックより取り出し

＋，－：集合間の演算の場合は，和集合，差集合を表す

探索条件 $E=\{e_{j1},..,e_{jp},\bar{e}_{k1},..,\bar{e}_{kq}\}$：$a_{j1},...a_{jp}$は正常，$a_{k1},...,a_{kq}$ は故障という条件

$|E|$：　E に含まれる要素の数

4．アルゴリズム

S0：スタックを空にする．$P_N=0$．$E=\phi$（空集合）をpush．

S1：スタックが空ならS3へ．そうでなければ，探索条件をpop．それをE_0とする．

S2：$E_0=\{e_{j1},e_{j2},..,e_{jp},\bar{e}_{k1},..,\bar{e}_{kq}\}$ とする．E_0を満足するネットワークの最小リンクをさがす．最小リンクがなければS1へ．$L_0=\{a_{m1},a_{m2},...,a_{mr}\}$ が最小リンク

Reliability of a Network with Capacity Constraint
Shigeru Yanagi, Tsutomu Asakami, Masafumi Sasaki
The National Defense Academy

であったとする．

$E=E_0+\{e_{m1},e_{m2},\ldots,e_{mr}\}$

$=\{e_{f1},\ldots,e_{fc},\bar{e}_{k1},\ldots,\bar{e}_{kq}\}$

とおく．

$P_N = P_N + p_{f1}\cdots p_{fc}\bar{p}_{k1}\cdots\bar{p}_{kq}$.

$|E|=n$（即ち，$c+q=n$）ならばＳ１へ．

$|E|<n$ならば，$E-E_0=\{e_{h1},\ldots,e_{hd}\}$ とし，以下のｄ個の探索条件を作る．

$E_1=E_0+\{\bar{e}_{h1}\}$,

$E_2=E_0+\{e_{h1},\bar{e}_{h2}\}$,

......

$E_d=E_0+\{e_{h1},\ldots e_{h(d-1)},\bar{e}_{hd}\}$

$E_1,E_2,\ldots,E_d$ をpush．Ｓ１へ．

ここで，$E_1,\ldots,E_d,\{e_{h1},\ldots,e_{hd}\}$は互いに排反な探索条件である．

Ｓ３：P_Nを出力．終了．

上記アルゴリズムの効率は大ざっぱに言って，最小リンクを捜す回数によって決定される．最小リンクを捜すアルゴリズムはラベリング法によって行う．上記アルゴリズムは［３］に比べてラベリング法へのアクセス回数を減らすことが可能である．また，アルゴリズム自体も本報告の方が単純である．次節でこれら２つのアルゴリズムの効率の比較を数値例を用いて行う．

５．数値例

図１は文献［３］で用いられている例である．信頼度は次式で与えられている．

$$P_N=Pr\{mf(s,t)\geqq 3\}^*$$

図２に前記アルゴリズムにより生成された探索条件の分岐木を示す．図中,Lの付いているノードは１レベル上の探索条件の下で発見されたリンクであり，Eの付いているノードは１レベル上の探索条件から新たに生成された探索条件である．この例の場合，ラベリング法へのアクセス回数は２２回である．一方，［３］のアルゴリズムを用いたときのアクセス回数は４７回であった．また，計算時間は，本アルゴリズムを適用すると，［３］に比べて，1/2.17となった．

６．あとがき

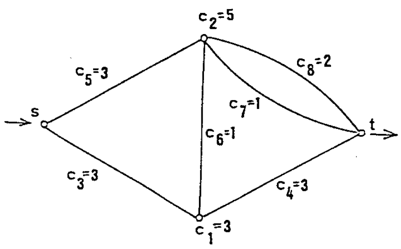
今回提案した方法を用いるとラベリング法のアクセス回数は一般に少なくなる．しかしながら，ラベリング法１回の計算に要する平均時間は逆に増加する可能性もある．この点に付いて多くの数値例を用いて検討をする必要がある．

参考文献

[1] K.K.Aggarwal,et al., Microelectronics and Reliability,22,1982, pp.337-340.

[2] H.Livni,Y.Bar-Ness,,IEEE Trans. Rel, R-26,1977,Apr.,pp.285-289.

[3] W.J.Rueger, IEEE Trans Rel. R-35, Dec.,1986,pp.523-528.



＊ノードｓ，ｔは故障しないと仮定する．ｓ，ｔはラベリング法適用時のみ用いる．

図１．ネットワークの例．

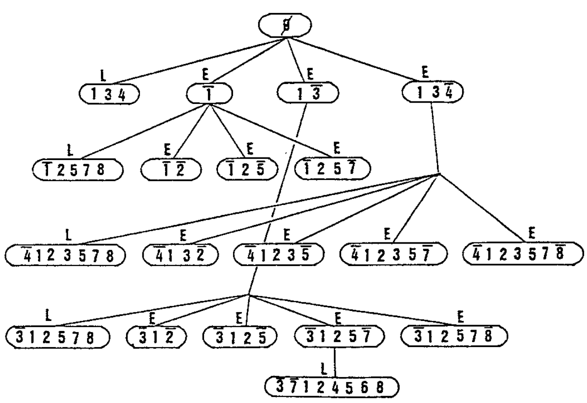


図２．探索条件の分岐木