4.3.4 **アニーリング法**（simulated annealing，SA 法）

近傍内の各解に解の良さに応じた遷移確率（よい解ほど移行し易い）を設定し，それに従って次の解をランダムに選ぶ.

遷移確率: 物理現象の**焼きなまし**（annealing）にアイデアを借りて，**温度**（temperature）と呼ばれるパラメータ t により調整.

改悪解であっても遷移する確率が正 → 局所最適解からの脱出.

メタ戦略の枠組のステップ II-C の移動戦略に対する工夫.

アニーリング法

ステップ 1 初期解 x を生成する．また，初期温度 t を定める.

ステップ 2 以下のステップ a, b および c を，ループの終了条件がみたされるまで反復する.

a $N(x)$ 内の解をランダムに一つ選び x' とする.

b $\Delta := \tilde{f}(x') - \tilde{f}(x)$ とする（x' が改悪解ならば $\Delta > 0$）.

c $\Delta \leq 0$ ならば確率 1，そうでなければ確率 $\mathrm{e}^{-\Delta/t}$ で $x := x'$（解 x' を受理）とする.

ステップ 3 反復の終了条件がみたされれば暫定解を出力して探索を終了．そうでなければ，温度 t を更新した後ステップ 2 に戻る.

温度 t: 通常探索の初期の段階では高めに設定（すなわち，ランダムな移動が生じやすい）．探索が進むにつれて徐々に小さく.

$t \to 0 \Longrightarrow$ アニーリング法の動作 → 通常の単純局所探索法.

ステップ 2 の受理確率

Δ: 評価関数値の差（正 $\Longrightarrow$ 改悪解）

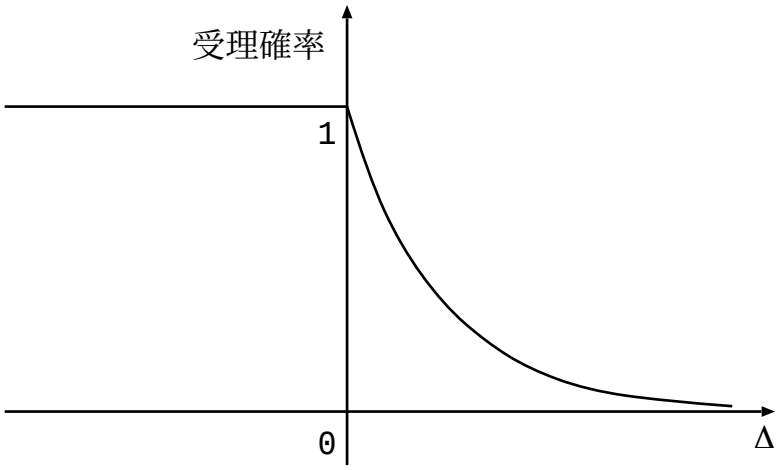

図 4.7 アニーリング法における解の受理確率 $\mathrm{e}^{-\Delta/t}$

1 次元の探索空間におけるアニーリング法の動作の様子

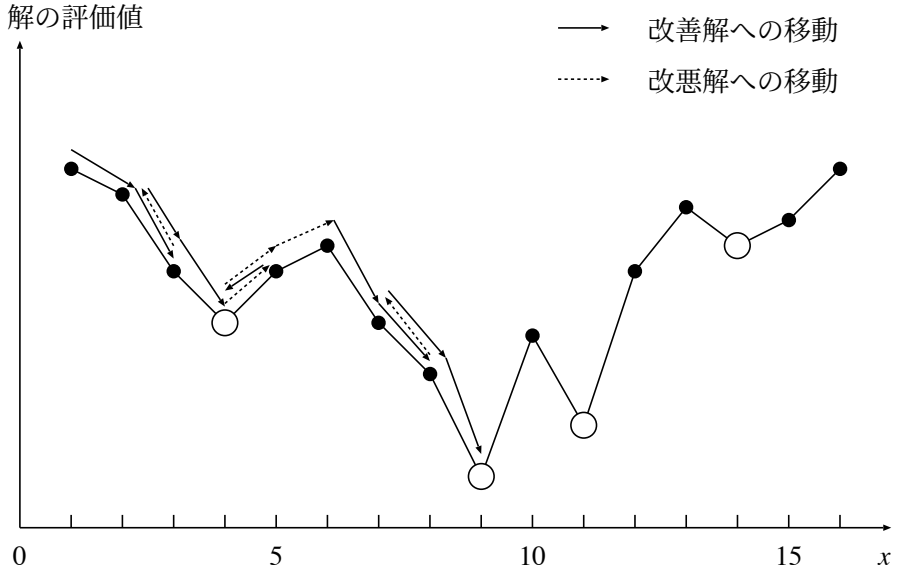

図 4.8 アニーリング法の進行の様子

確率的に改悪解へ移動 → 改善解があるのに逆戻りすることも.

解の評価値に応じた遷移確率 → 大局的にはよりよい解へ探索が進む.

アニーリング法の具体的実現

- ステップ 1 の初期温度の定め方,
- ステップ 2 のループの終了条件,
- ステップ 3 の温度の更新方法と探索の終了条件,

を定める必要．以下，これらの基本的な方法を紹介.

初期温度の定め方

探索の初期の段階で，生成した解の中で受理されるものの割合が p_{init} 程度となるように初期温度 t を設定（$0 < p_{\text{init}} < 1$ はパラメータ）.

注: 初期温度 t を直接調整 → 問題例ごとに適切な値が異なる.

ループの終了条件

基本ルール: ループを $\alpha_{\text{loop}} \cdot |N(x)|$ 回反復したら終了.

追加ルール: ループの中で受理された解の数が $\alpha_{\text{cut}} \cdot |N(x)|$ に到達したら，$\alpha_{\text{loop}} \cdot |N(x)|$ 回の反復が終るよりも前であっても終了.

$\alpha_{\text{loop}} > 0$ と $\alpha_{\text{cut}} > 0$ はパラメータ.

t が大 → ほとんどの解が受理 → 追加ルールはそのような時点での探索を早めに切り上げる効果.

温度の更新方法（冷却スケジュール（cooling schedule）と呼ぶ）

$t := \beta_{\text{temp}} \cdot t$ と更新（$0 < \beta_{\text{temp}} < 1$ はパラメータ）

→ **幾何冷却法** (geometric cooling)．簡単かつ実用的とされる.

SA 法の動作: 温度高 → ランダムウォーク; 温度低 → 単純局所探索.

温度がその中間 → アニーリング法の効果 → 以下の再アニーリング.

再アニーリング: 暫定解が見つかった時点の温度 t_{found} を記憶．暫定解がしばらく更新されないとき温度を $t := t_{\text{found}}$ と一旦高く.

探索の終了条件

ステップ 2 のループの中で解の受理頻度が p_{freeze} 以下となる反復が r_{freeze} 回続いたら終了（温度が十分低くなったと判定）.

p_{freeze} と r_{freeze} $(0 < p_{\text{freeze}} < 1 \leq r_{\text{freeze}})$ はパラメータ.

考察

ステップ 2-a: 近傍内の解を試行ごとに独立にランダムに選択.

簡単のため，このルールがしばしば利用される.

しかし，現実には，この方法は効率が悪い.

理由: 独立なランダム試行では同じ解を繰り返し選ぶ確率が高い.

例: 近傍 $N(x)$ の中に改善解が丁度一つあるという状況を考える.

解をランダムに選ぶ試行を独立に $k|N(x)|$ 回繰り返したときに改善解を 1 度も選ばない確率: $(1 - 1/|N(x)|)^{k|N(x)|} \simeq \mathrm{e}^{-k}$ （結構大）.

k	1	2	3	4
e^{-k}	0.37	0.14	0.050	0.018

すなわち，近傍のサイズの数倍程度の試行を繰り返しても，望ましい解が一度も選ばれない確率は無視できないほど大きい.

このような問題点を回避する方法の一例:

近傍内の全解を一度ずつ探索するランダムな順序をあらかじめ設定し，この順序に従って解を抽出.

閾値受理法 (threshold accepting) と **大洪水法** (great deluge algorithm):

アニーリング法に類似の方法. 遷移確率や遷移のルールがより単純.

4.3.5 **タブー探索法**（tabu search, TS 法）

基本方針: 近傍 $N(x)$ 全体の中で x 以外の最良解を次の解とする.

（近傍サイズが大きい場合は $N(x)$ の一部に限定する場合もある.）

→ 現在の解 x が局所最適解であっても他の解への移動が強制.

現在の解 x が局所最適解 & N が対称

→ x から $x' \in N(x)$ へ移動後 $N(x')$ 内の最良解 $= x$ の可能性大.

i.e., **サイクリング** (cycling).

タブーリスト (tabu list): 移動が禁止される解集合 T.

短期メモリ (short term memory, recency based memory) とも呼ぶ.

探索は $N(x) \setminus (\{x\} \cup T)$ 内の最良解へ移動 → サイクリングを回避.

メタ戦略の枠組のステップ II-C の移動戦略に対する工夫.

タブー探索法（基本構成）

ステップ 1 初期解 x を生成する．タブーリスト T を初期化する.

ステップ 2 $N(x) \setminus (\{x\} \cup T)$ の中で最も望ましいと考えられる解 x' を見つけ，$x := x'$ とする.

ステップ 3 終了条件がみたされれば暫定解を出力して探索を終了する．そうでなければ，T を更新した後ステップ 2 に戻る.

注: タブー探索法では，さらに，特定の変数を変更した頻度や，特定の変数がある値をとり続けた期間の長さなど，探索解の特徴を長期間に渡り記憶しておくことにより（**長期メモリ**（long term memory)），未探索の領域へ探索を方向づけようとする手法を組合せて用いることが多い．講義では短期メモリに基づく基本構成のみを解説.

タブーリストの構成

タブーリスト T: 最近探索した解が含まれるように動的に制御.

T に最近探索した解を直接記憶 →

- 生成した解が T に含まれるかどうかの確認に時間がかかる.
  （データ構造を工夫すれば高速な確認も可能だが実現が面倒.）
- サイクリングを防ぐのに十分な効果が得られない.

よって，解を直接記憶する方法は通常用いない.

代表的なルール: 最近の近傍操作において移動の前と後で値の変わった変数や，変数とその値のペアなどを $\tilde{T}$ として記憶.

- $\tilde{T}$ 内の変数の値を変更することを禁止する，あるいは
- $\tilde{T}$ 内の変数の値が変更前の値に戻ることを禁止する.

利用される移動の特徴: **属性**（attribute）.

$\tilde{T}$: 禁止解の集合と同様の役割を持つのでタブーリストと呼ばれる.

上記の禁止規則を探索の間中保持 → 移動できる解がなくなる.

→ 一つの属性は $\tilde{T}$ に入ってから t_{tabu} 回反復の後 $\tilde{T}$ から除く.

（ステップ 2 における解の移動 1 回を 1 反復と数える.）

t_{tabu}: パラメータ. **タブー期間**（tabu tenure）と呼ばれる.

タブー探索法の性能は，タブー期間の値 t_{tabu} に対して非常に敏感.

属性に基づくタブーリスト

→ 問題や近傍の構造を考慮して構成する必要あり.

タブーリストの構成例

最大充足可能性問題で1反転近傍を用いた場合.

禁止規則1 移動の前と後で変数 v_i の0-1割当が変更されたとき，添字 i をタブーリストに記憶する．タブーリストに含まれる全ての添字 i について，v_i の値の変更を禁止する.

この場合，属性は各添字.

2反転近傍を用いた場合

禁止規則2 移動の前と後で0-1割当が変更された変数の添字全てをタブーリストに記憶する．すなわち，v_i の値が変更されたときは添字 i を，v_i と v_j の2つの値が変更されたときは添字 i と j の両方をタブーリストに加える．そして，タブーリストに含まれる全ての添字 i について v_i の値の変更を禁止する.

禁止規則3 移動の前と後で0-1割当が変更された変数の添字の集合をタブーリストに記憶する．すなわち，v_i のみの値が変更されたときは集合 $\{i\}$ を，v_i と v_j の2つの値が変更されたときは集合 $\{i, j\}$ をタブーリストに加える．そして，タブーリスト内の添字集合に対し，0-1割当が変更される変数の添字集合が一致する移動を禁止する.

禁止規則2: タブーリスト内の変数を一つでも含む変更を禁止.

禁止規則3: 変数のペアに対し同じペアを変更することを禁止.

通常，禁止規則2のようなルールでは制限が強すぎる.

順列の集合が探索空間，交換近傍 N_{swap} を用いた場合

一つの k に対して $\sigma(k)$ の取り得る値が多数あることを考慮した規則:

禁止規則 4 移動の前後で順列 σ の k 番目と l 番目の要素が交換されたとする．このとき，添字と要素の（順序つき）ペア $(k, \sigma(k))$ と $(l, \sigma(l))$ をタブーリストに記憶する．そして，タブーリストにペア (k, i) が存在する場合には，順列の k 番目の要素が i となる移動を禁止する．

この場合の属性は，添字と要素のペア．

巡回セールスマン問題における 2-opt 近傍を用いた場合

移動によって枝が解に加えられることと枝が解から除かれることが対称的でないことを考慮した規則:

禁止規則 5 移動によって解に加えられた枝をタブーリストに記憶する．そして，タブーリストに入っている枝を解から除くような移動を禁止する．

禁止規則 6 移動によって解から除かれた枝をタブーリストに記憶する．そして，タブーリストに入っている枝を解に加えるような移動を禁止する．

この場合の属性は枝．

禁止規則 5 と 6 では 5 のほうがより強い制約 →

組合せて用いる場合，禁止規則 5 に対する t_{tabu} よりも禁止規則 6 に対する t_{tabu} のほうを長めにするような配慮が必要．

タブーリストの計算法

属性のリストをそのまま保持し，解を探索するたびにそのリストを走査するのでは効率が悪い.

禁止規則 1 の実現例

大きさ n（n は変数の個数）の配列 $\mu = (\mu_1, \ldots, \mu_n)$ を用意.

全ての j に対して $\mu_j := -\infty$ と初期化.

探索が始まった時点からの反復回数 c をカウント.

反復回数 c における解の移動で v_j の値が変更 $\rightarrow$ $\mu_j := c$ と更新.

変数 v_j の値を変更する移動が禁止されているかどうかの確認:

$$(\text{そのときの反復回数}\, c) - \mu_j \leq t_{\text{tabu}}$$

が成立すれば禁止されていると判定.

t_{tabu} の値に関わらず $\mathrm{O}(1)$ 時間で判定可能.

特別選択基準（aspiration criteria）

タブーリストによって禁止されている解 $x' \in N(x) \cap T$ でも,

- 解 x' を採択してもサイクリングが起こらない,
- 解 x' を採択することに十分意味がある,

と判断される場合,（タブーを犯して）その解への移動を実行.

これを判定する基準 $\rightarrow$ 特別選択基準.

代表的な特別選択基準の例:

- x' は実行可能解かつ目的関数の値 $f(x')$ が暫定値を更新する.
- 評価関数の値 $\tilde{f}(x')$ がこれまで探索したどの解よりも小さい.