四色問題の証明
Formal proof of the Four color theorem

加藤　一郎
Ichiro KATO
e-mail : ***kato@my.zaq.jp***

要約

四色問題とは、「平面上の如何なる地図も、隣接する領域を塗り分けるには四色あれば十分である」ということを示す問題である。この問題は、「放電法」と呼ばれる手法を発展的に利用し、人手による実行が事実上不可能なほど複雑なプログラムによって 1976 年に証明された。その後も断続的にアルゴリズムやプログラムの改良が行われ、現在では四色問題は解決しているとみなされ、「四色定理」と呼ばれるようになった。しかし、その証明は計算機による済し崩しの演算による解決には違いなく、現在でもコンピュータを利用せずに済ませられる証明は得られていない。本稿は論理的な着色プロセスを示すことで、あらゆる地図が四色で塗分け可能なことを、計算機を用いずに証明するものである。

キーワード ： 四色定理、グラフ理論、色スワップ、連鎖ルート、分断ルート、素系
Keywords **:** Four Color Theorem, Graph Theory, Color Swap, Chain Route, Dividing Route,Elementary system

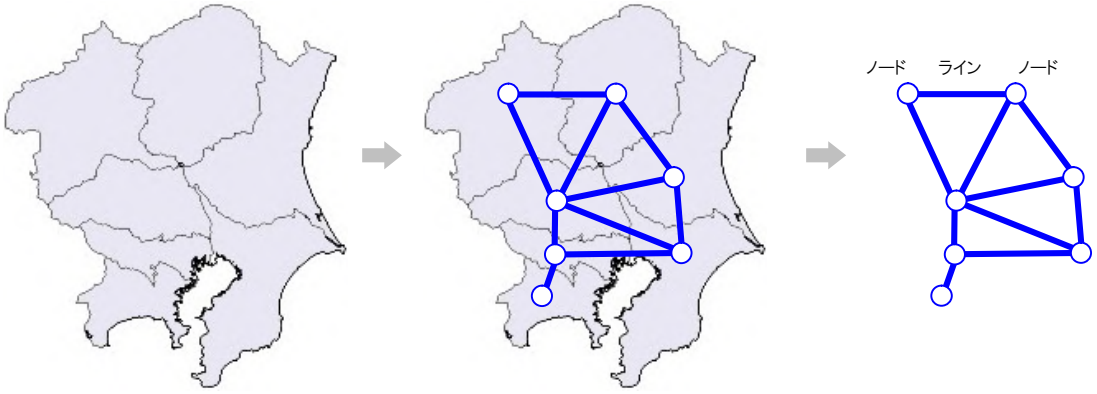
1. はじめに

本稿は、以下のアイディアを結び付けるとこで、あらゆる地図が四色で塗分けられることを証明する。

- 地図をグラフ化し、そこに補助線を加えることで三角ネットのみの還元可能な構成を与える
- ノード接続とライン接続という単純な手続きを積み上げて、全体を着色するプロセスを確立する
- 2色のスワップ特性を考察し、分断ルートという概念を用いて同色ライン接続を解消する
- ルート変更の輻輳状態から、分断ルートの解が必ず得られることを示し、四色問題を証明する

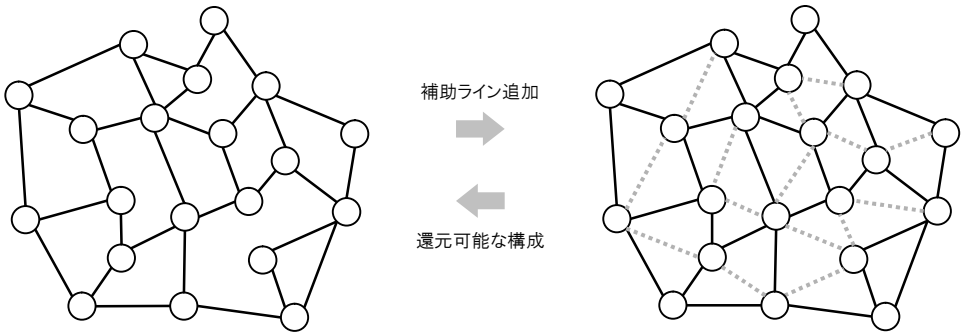
2. 平面グラフ化

極めて一般的な手法ではあるが、まずは地図の平面グラフ化を実施する。以下の例のように、地区ごとにノードを与え、地区間の境をラインで接続する。当然のことながら、これらのラインは交差することは無い。このノードのそれぞれに色を割り当てていくことになるが、ライン接続されたノード同士は異なる色を与えなければならない。この条件下において、どのような地図であろうとも4色で塗り分けられることを主張するのが、四色問題である。

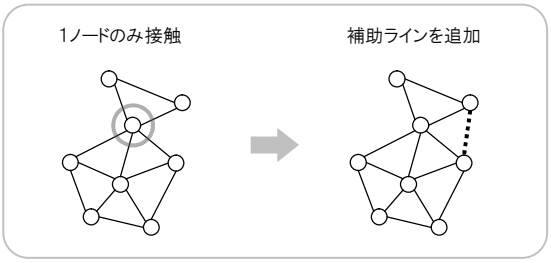


3. 補助ライン

これも一般的な手法ではあるが、以降の分析のために一つの工夫を施す。得られたグラフの各ノードに補助ラインを追加し、すべて三角のネット網となるようにする。この処理はノードの隣接状態を複雑にし、4色の塗分け条件がオリジナルより厳しくなる。しかし、塗分けパターンは逆に単純化されるため、トポロジー分析において有利に働く。三角ネット網の状態でグラフ全体が4色に塗り分けられたら、追加した補助ラインを除去することで、元の地図のグラフが復元する。つまり、三角ネット網で塗分け可能ならば、元の地図も塗り分けられることが保証される。

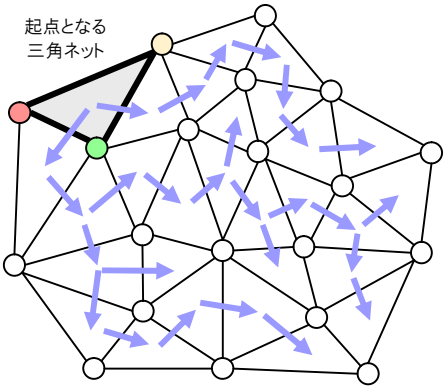


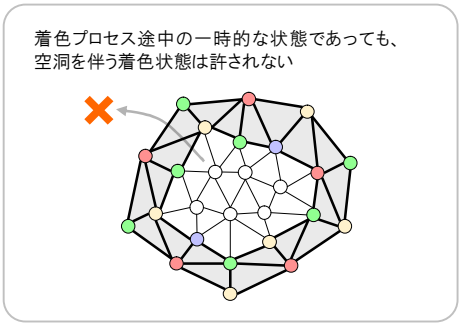
なお、1ノードのみ接触している三角ネットが存在する場合は必ず補助ラインを追加し、2 ノード以上の接触状態にしておく。これにより、着色プロセスの処理パターンがより簡素化する。



4. 着色プロセス

着色プロセスは、まず起点となる任意の三角ネットの着色を行う。次に着色済みのノードに対して、「ノード接続」「ライン接続」のいずれかの手段により、順次着色を拡散していくことで、最終的にグラフ全体を着色する。なお、着色済みの三角ネットに隣接してさえいれば着色順序や接続手段は任意であるが、あくまでも稠密に実施し、空洞を伴う着色状態（ドーナツ状態）に陥らないように留意しなければならない。



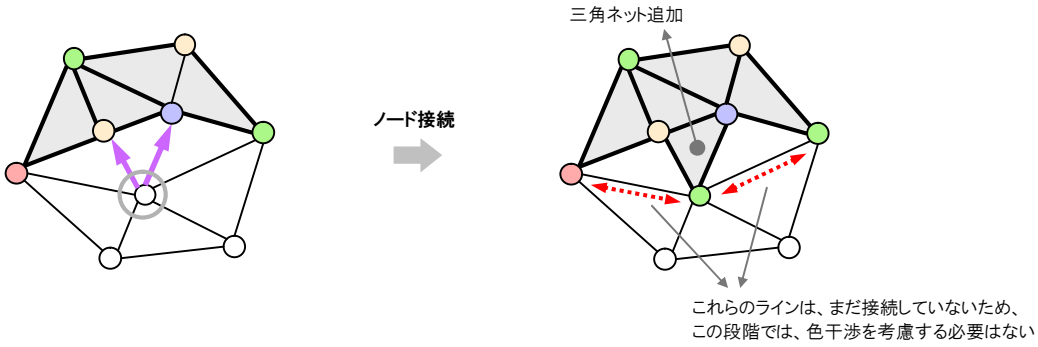


5. 接続と着色

着色の際に使用する2つの接続、「ノード接続」と「ライン接続」について解説する。

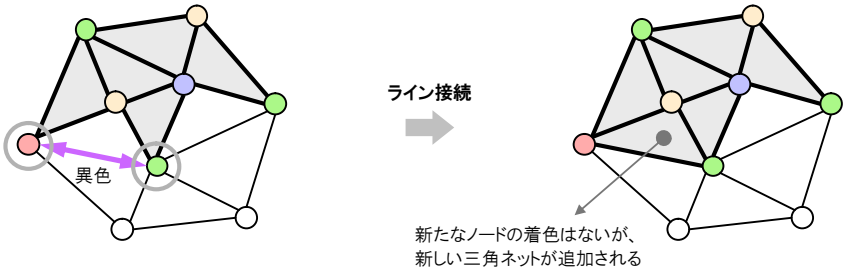
ノード接続

着色済み、かつ接続済みの 2 つの隣接ノードに、新たに1ノードを接続して着色し三角ネットを追加する。追加されるノードは、接続先の2つのノードと異なる色で着色できることが保証される。なお対象とする接続先ノードは、あくまでも2ノードのみであり、それ以外の着色済みノードが存在していても、そこには三角ネットを追加せず、それらのノードとの色干渉もこの段階では考慮する必要はない。その解決は、後続のライン接続に委ねられる。

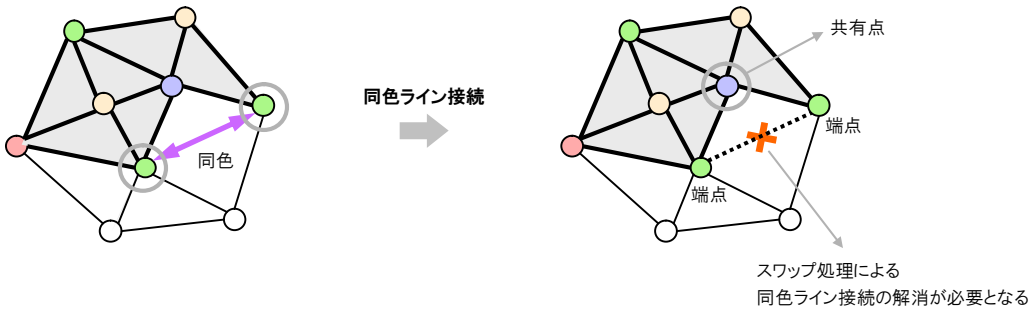


ライン接続

新たに着色するノードはなく、着色済みノードの色が干渉しないことを確定する処理である。確定されると新たな三角ネットが追加される。ライン接続の際、接続する2ノードの色が異なる場合は、単純な接続のみでよい。



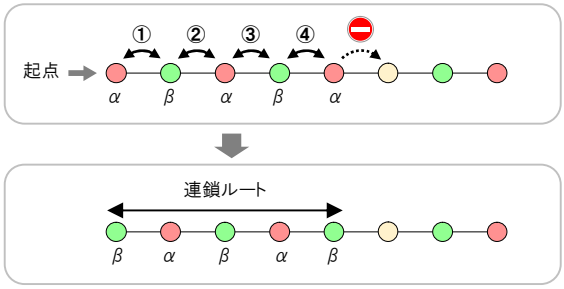
一方、ライン接続する2ノードが同色の場合は、周辺のノード色を組み換えて2ノードを異なる色にする必要がある。このライン接続を「同色ライン接続」と呼び、色の組換え処理を「スワップ」と呼ぶ。同色ライン接続において、接続を行おうとしている2ノードを「端点」と呼ぶ。また、両端点から接続された同一のノードが必ず1つ存在し、それを「共有点」と呼ぶ。



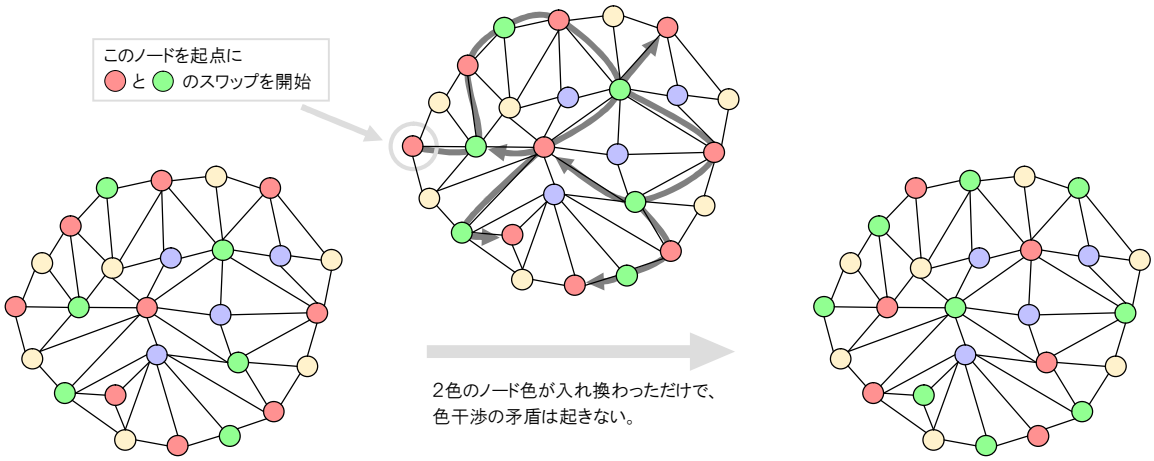
これらの接続を順次実施していけば、グラフ全体が着色できることは自明であるため、同色ライン接続を確実に解消できる手順を示すことができれば、四色問題は証明されたことになる。

6. スワップ

スワップは、起点となるノードの色（α）を他の色（β）に変更することから始める。その際、起点ノードに（β）のノードが接続されていた場合（β）の干渉が起こる。そこで、干渉するノード（β）を（α）に色変更することで干渉を解決する。しかし、更にその先に（α）のノードがあった場合、そこで再び干渉が起こるため、そこでも色を変更する必要がある。このようにスワップは、起点ノードから（α）と（β）が交互に存在しているルート上の色を連鎖的に入れ換える処理となる。このルートを「連鎖ルート」と呼ぶ。スワップは、接続されているノードのすべてが（α）（β）以外の色となる箇所に到達した時点で停止する。



スワップは、起点からの色の入れ換えが、分岐やループを伴いながら広範囲に及ぶ可能性がある。しかし、スワップ前のグラフが色干渉していなければ、たとえ分岐やループを発生したとしても、スワップ後も色干渉の破綻は起きない。



7. 分断ルート

片側の端点を別の色に変更すれば、同色ライン接続は解消するが、着色済みのノードとの色干渉を避けるためには、スワップによる色変更を実施しなければならない。しかしもしスワップの連鎖ルートが相手の端点に到達してしまったら、共に同じ色に変更する結果となり、同色ライン接続の解消にはならない。よって同色ライン接続を解消するには、相手の端点に到達しないスワップが必要となる。そして、その状況を作り出す手段もまたスワップを利用する以外にない。

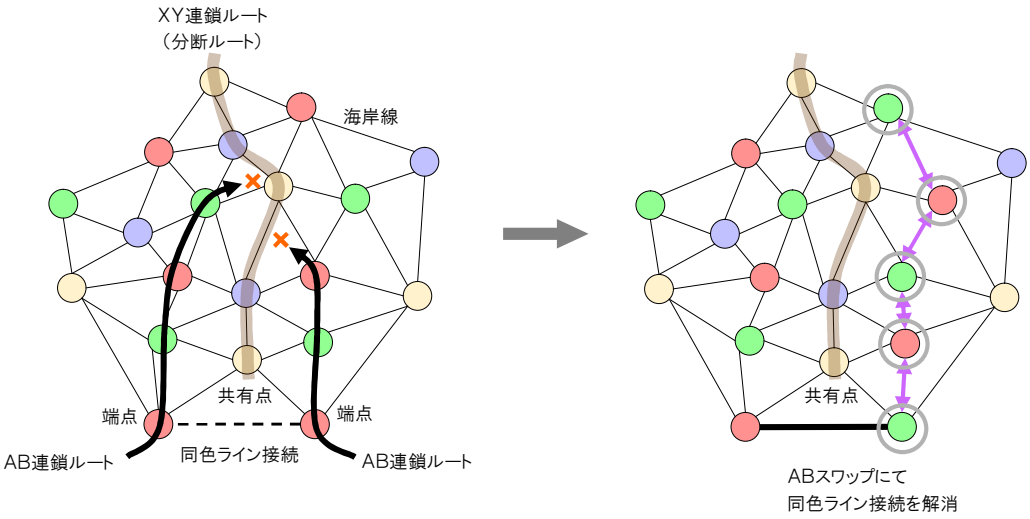
端点色と対をなすその他3色のスワップのどれか1つでも、相手端点への未達スワップがあれば、その色のスワップで同色ライン接続を解消できる。しかし3色のスワップすべてが導通してしまったら、単純なスワップ処理では解決できない。端点間を導通する3色の連鎖ルートがそれぞれ複数あり、それらが複雑に絡み合っているグラフでは、更にその解を得るのが困難になる。

ここで、1つの定理を準備する。

定理 ： 排他的な色の組合せで構成される2つの連鎖ルートは交差しない

例えば、与えられた 4 色を A,B,X,Y とした時、AB 連鎖ルートと、それと排他的な色の組合せである XY 連鎖ルートは交差しない。何故ならば、交差するには交差点であるノードの色は同一である必要があるが、それぞれの連鎖ルート上には異なる色のノードしか存在しないためである。

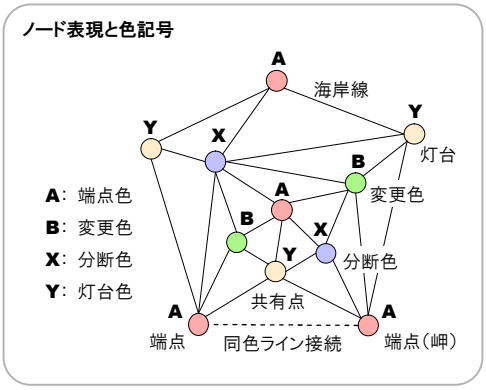
この定理を使うと、その時点で着色されている輪郭上のいずれかのノードから共有点に向かって XY 連鎖ルートを開通させれば、端点間の AB 連鎖ルートが導通しない状況を実現させることができる。この XY 連鎖ルートを「分断ルート」と呼ぶ。分断ルートはその1本で、端点間を導通する AB 連鎖ルートのすべてを、一挙に除去することができる。よって、もし分断ルートを確実に開通させる手続きを示すことができれば、同色ライン接続の解消は保証され、ひいては四色問題を証明できたことになる。

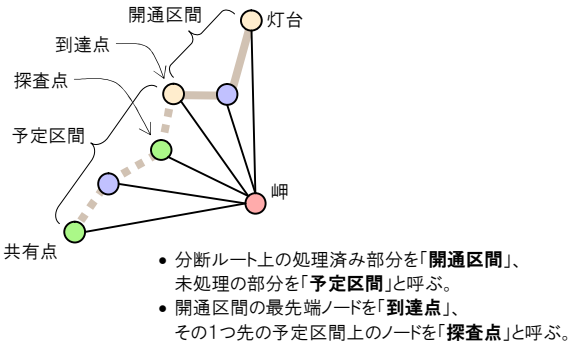


8. 各種用語定義

スワップを活用して分断ルートを開通させ、それをもって同色ライン接続を解消するというプロセスの見通しを良くするために、以下の各種用語定義を与える。

- 海岸線　：　着色済みノードで最外殻を形成する輪郭ライン
- 岬　　：　色変更を試みるどちらか一方の端点
- 灯台　　：　岬に直接接続されている、共有点と反対側の海岸線ノード
- 変更色　：　最終的に岬を変更する色
- 分断色　：　灯台色と対となる分断ルートの要素色





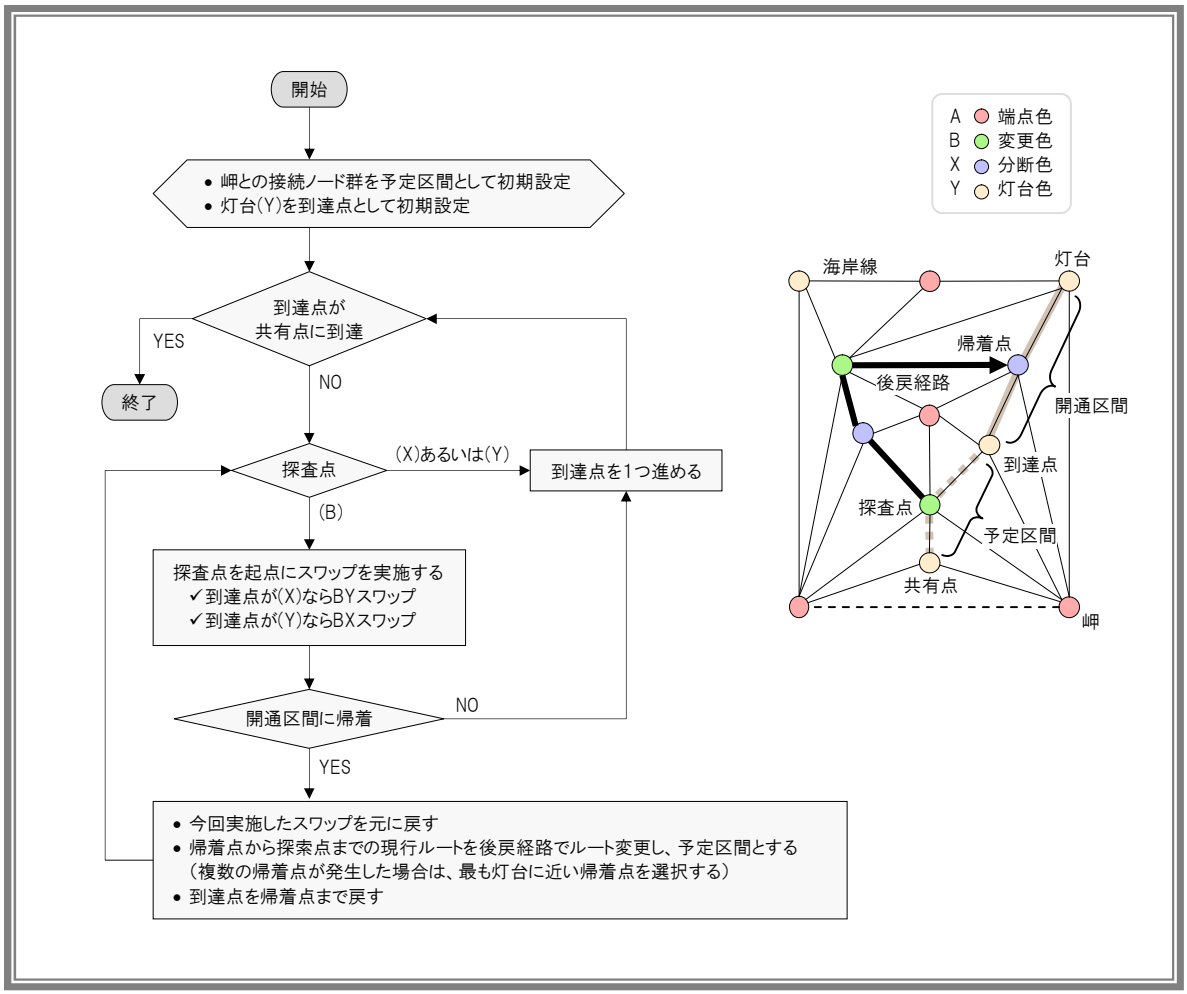
- 分断ルート上の処理済み部分を「**開通区間**」、未処理の部分を「**予定区間**」と呼ぶ。
- 開通区間の最先端ノードを「**到達点**」、その1つ先の予定区間上のノードを「**探査点**」と呼ぶ。

9. 分断ルート開通手順

端子間を導通するAB連鎖ルートを一掃するため、XY分断ルートを開通させる手順を示す。

手順の大まかな流れは、まず岬に直接接続されているノードを予定区間とし、灯台を到達点の初期位置にする。次に灯台から共有点に向かって XY 連鎖ルートを構築しながら開通区間を伸ばしていく。探査点が B に到達した場合は、スワップ処理で X または Y に色変更しながら到達点を前に進める。もしスワップが構築済みの開通区間に後戻りした場合は、到達点を帰着点に戻すと共に、後戻経路でルート変更する。この手続きを繰返して到達点を共有点まで進め、分断ルートを開通させる。

なお、初期設定時の予定区間は端点と接続されているため端点色(A)は存在せず、端点色(A)に絡むスワップ処理も存在しないため、端点色(A)は分断ルート上に現れない。色変更も実施されないため端点色(A)のノードは固定化されている。

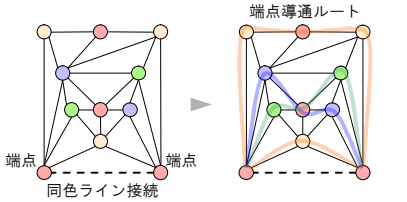


後戻経路を検出しルート変更の必要が生じる局面は、その時点で設定されている予定区間では完全な分断ができないことを、アルゴリズムそのものが検知したとも考えられ、後戻経路でルート変更することでより適切な予定区間が与えられることを示唆している。後戻経路を生み出す連鎖ルートは、海岸線を走行して灯台に達する場合もあれば、開通区間をよぎって再び開通区間上の別のノードに達する場合もある。そのため、ルート変更は極めてダイナミックに実施される。しかし灯台と共有点という両極のノードは不動であるため、最終的に得られる分断ルートが両端点を分断することは保証されている。

開通した分断ルートは XY 連鎖ルートとなり、両端点の AB 連鎖ルートを文字通り分断する。よって、どちらか片方の端点を起点に AB スワップを実施すれば、連鎖はもう一方の端点に到達せず、結果として同色ライン接続は解消する。したがって、如何なる場合でも本手順が分断ルートを導けることを示すことができれば、四色問題を論理的に証明したことになる。

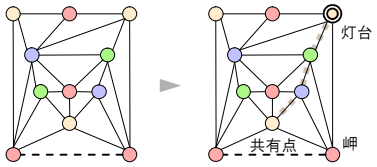
10. 同色ライン接続の解消例

提示した分断ルート開通手順に従って、同色ライン接続の解消を行う具体的な実施例を示す。



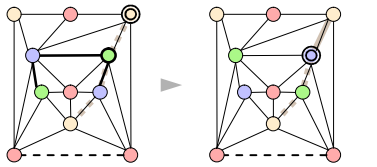
① 同色ライン接続が発生。
両端点間に3色すべての導通ルートが存在しているため、単純なスワップでは解消しない。

- A 端点
- B 変更色
- X 分断色
- Y 灯台色

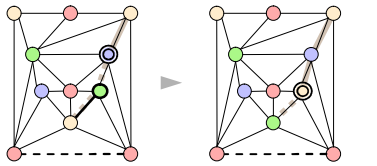


② 岬に接続されたノード群を分断ルートの予定区間とし、到達点の初期位置として灯台を設定する。

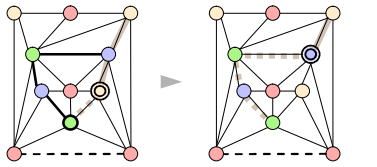
- ✓ 分断ルート （予定区間） （開通区間）
- ✓ 到達点
- ✓ 探査点
- ✓ スワップ対象



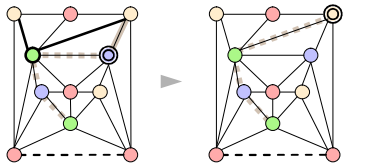
③ 探査点が B であり、到達点が Y であるため、探査点を起点に BX スワップを実行。
後戻りは発生しないため、スワップ後、到達点を1つ進める。



④ 探査点が B であり、到達点が X であるため、探査点を起点に BY スワップを実行。
後戻りは発生しないため、スワップ後、到達点を1つ進める。

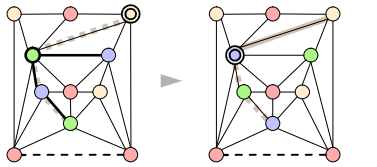


⑤ 探査点が B であり、到達点が Y であるため、探査点を起点に BX スワップを実行。
後戻りが発生するため、今回のスワップを元に戻し、後戻経路で分断ルートをルート変更する。
到達点を帰着点まで戻す。



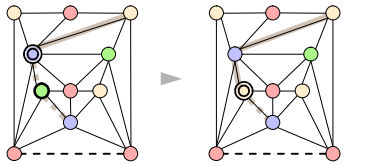
⑥ 探査点が B であり、到達点が X であるため、探査点を起点に BY スワップを実行。
後戻りが発生するため、今回のスワップを元に戻し、後戻経路で分断ルートをルート変更する。
到達点を帰着点まで戻す。

※ このように、到達点が灯台まで戻る場合もある。



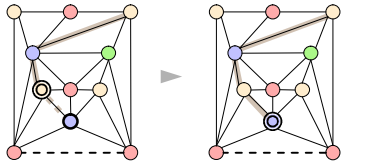
⑦ 探査点が B であり、到達点が Y であるため、探査点を起点に BX スワップを実行。
後戻りは発生しないため、スワップ後、到達点を1つ進める。

※ 帰着判定の開通区間は、その時点の最新の分断ルート上である。
ルート変更前の分断ルートに到達しても、後戻り発生と判定しない。

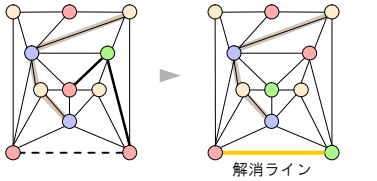


⑧ 探査点が B であり、到達点が X であるため、探査点を起点に BY スワップを実行。
後戻りは発生しないため、スワップ後、到達点を1つ進める。

※ 起点にスワップ先のノードがない場合でも、自身の色変更だけは行う。



⑨ 探査点が X のため、そのまま到達点を1つ進める。
到達点が共有点に達したため、分断ルート開通の処理は完了。

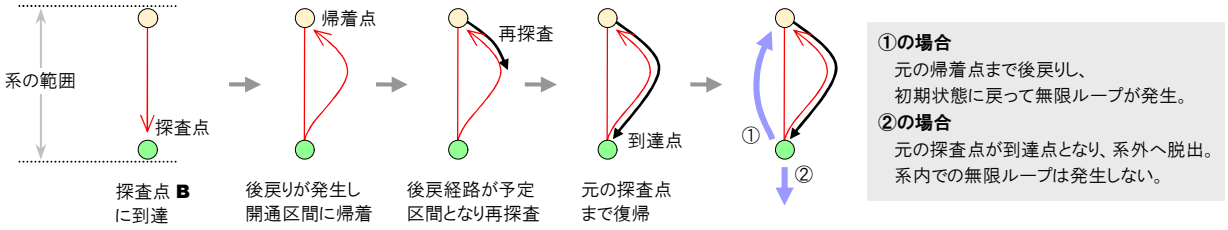


⑩ 灯台から共有点に至る分断ルート(XY 連鎖ルート)が開通していることが確認できる。
岬を起点に AB スワップを実行し、同色ライン接続は解消される。

11.分断ルートの存在証明

提示した分断ルートの開通手順が如何なる場合でもその解を導けることを示すことで、四色問題を論理的に証明する。分断ルートの解が必ず存在するということは、灯台を初期位置とする到達点が、必ず共有点に達することと等価である。到達点が更新される最も単純な状況は XY が交互に並んだ予定区間であり、その場合はスワップを行わなくとも到達点を前に進めることができる。探査点が B を検知した際スワップが発生するが、このスワップで後戻りが生じない場合も、そのスワップ後に到達点を前に進めて開通区間を伸ばすことができる。よって、検証すべきは、ルート変更が絡む場合に絞り込むことができる。与えられたグラフは有限のノード数で構成されており、本手順の停止条件は共有点への到達のみであるため、永遠に共有点に達することができないグラフは、手順の過程で無限ループに陥るパターン以外にはない。したがって、ルート変更の起点となる探査点 B において、後戻りの無限ループが発生せず、到達点が共有点に向けて必ず前進することを証明すれば良い。

今、探査点がBに到達し、そこから後戻りが発生して開通区間上のある帰着点（XあるいはY）に達したとしよう。この時、後戻経路によるルート変更が起こり、到達点は帰着点まで戻されて、再び元の探査点に向けて新しく設定された予定区間を前進する。このプロセスを1つの「系」と考える。この系が無限ループを起こさず、最終的に元の探査点が新しい到達点となり系外へ脱出することを示すのが、分断ルート存在証明のゴールとなる。

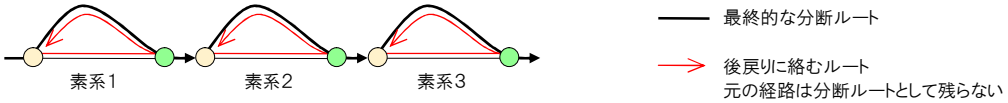


（注）図中の帰着点は **C** としているが **X** の場合もある。なお帰着点は開通区間上のため **B** はあり得ない。
また、実際には帰着点と探査点の間に他のノードが存在しているが、図中は省略している。

グラフ全体は複数の系が複雑に輻輳している状態だと言える。したがって、輻輳された全体の系が無限ループを起こさないことを示す必要がある。そこでまず、最も小さな系の要素を考えてみよう。これを「素系」と呼ぶ。素系とは、その中に他の系を含まない系である。よって、素系は後戻り後の復帰探査中に更に後戻りが発生することは無い。ここで「素系は無限ループを起こさない」と仮定して系の輻輳状態を、直列、循環、再帰の3つの状態に分類して分析してみる。

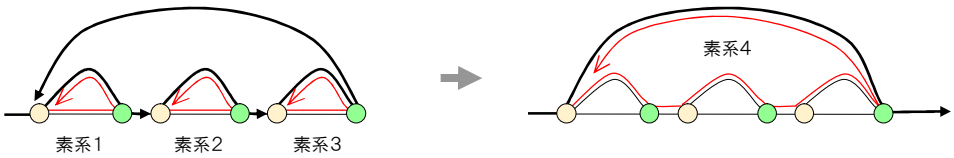
(1)直列

素系が順次接続された状態である。個々の素系が無限ループを起こさず次のステップに進むことを仮定しているため、直列に接続された系全体も無限ループを発生することはない。



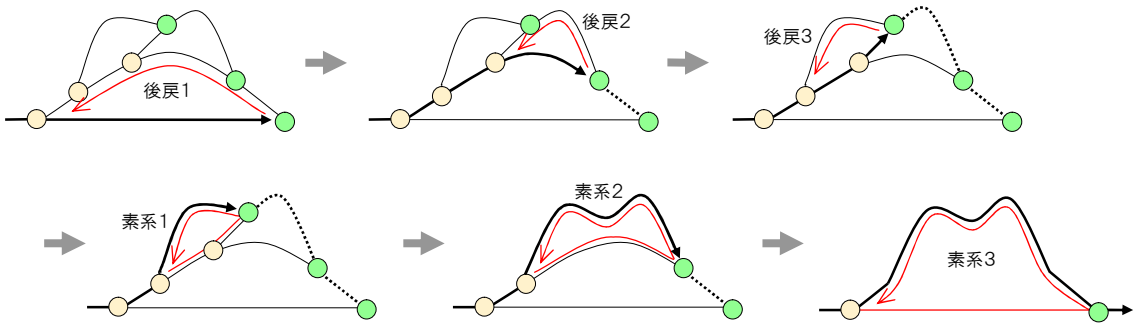
(2)循環

ある素系が以前の素系の帰着点に戻って循環する状態である。一般的な循環プロセスの場合、この状況に陥ると無限ループとなるが、分断ルート探査は後戻りが発生するとルート変更が起こるため一般的な状況とは異なる。本アルゴリズムでは個々の小さな系の経路はすべてクリアされ、大きな系の経路だけが残って、結果として1つの大きな素系となる。

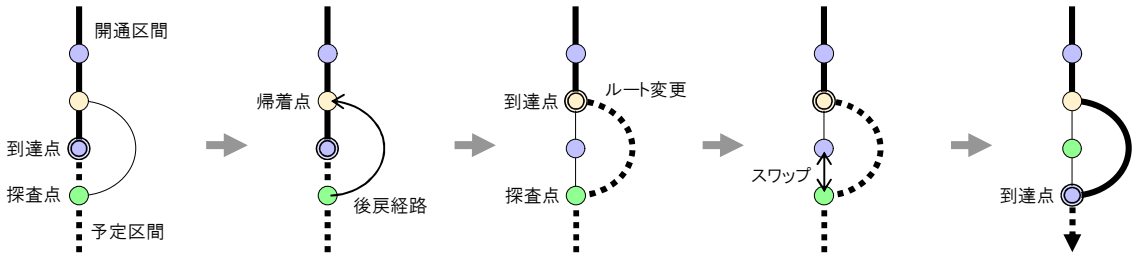


(3)再帰

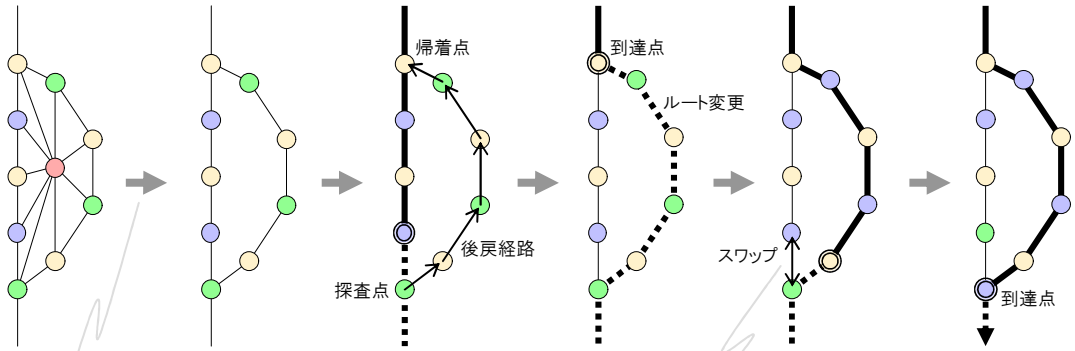
後戻りプロセス中に再帰的に後戻りが生じる状態である。この場合、階層の最終段が素系となることは自明である。最終段の素系は無限ループを起こさないため、その1つ前段の階層から見れば単純な進行ルートと等価である。よって、その2つの後戻りプロセスは1つの素系に統合することができる。これを順次繰り返すことで、階層化された再帰的な後戻りは1つの大きな素系となる。



以上の分析により、素系は無限ループをしないという仮定の下では、どのような輻輳状態であろうとも全体系も無限ループしないことが示された。よって、系の根源要素となる素系が無限ループに陥らないことを示せば、分断ルートは必ず開通することになる。最も単純な素系は後戻経路にノードが存在せず、経路の距離も最小の場合である。この場合、ルート変更後の再探査ですぐに元の探査点に戻る。ここでのスワップは必ず成功し、到達点はこの系から脱出できることは自明である。



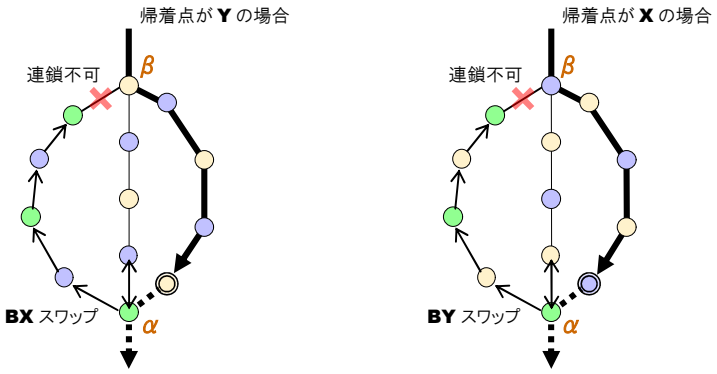
後戻経路に複数のノードが存在する場合でも、到達点の脱出パターンは変わらない。結局、再度元の探査点に戻ってきた際、そのスワップを通じて再び元の帰着点まで後戻りしない限り、無限ループは生じないことがわかる。



端点色 A はスワップ処理や分断ルート上のノードに現れず、
分断ルート開通手順に一切関与しないため、検証の際には省略可能である。

このスワップが元の帰着点に戻る前に停止する限り、
無限ループは生じない。

今、後戻りが発生した時点の探査点をα、その後戻経路が帰着した点をβとする。到達点はβまで戻され、αに向けた再探査がルート変更された経路に沿って開始される。それがαまで戻ってきた際、スワップが連鎖して再びβまで戻るためには、その後戻りを実現するための連鎖ルートがα ββ間に存在しなければならない。例えば、帰着点が Y であった場合、ルート変更後の予定区間は BX スワップでαに戻っていくことになるが、そのスワップがそのまま連鎖してβに再び後戻りするためには、αからβまで BX 連鎖ルートが接続されている必要がある。しかし、BX 連鎖ルートでは帰着点 Y を連鎖させることはできない。帰着点の手前で BX スワップが停止するため、スワップ処理は遂行され、到達点はこの素系から脱出する。分断ルートを構成する X と Y は対称性を持っているため、帰着点が X の場合も同様のことが言える。もし帰着点が X の場合は、ルート変更後の予定区間は BY スワップでαに戻るため、そのスワップをβに向けてそのまま連鎖させても、帰着点 X までは連鎖できない。この場合も、やはり到達点は素系を脱出する。よって、無限ループを発生させるために必要なαからβへの後戻経路は、素系内で実現することは不可能であることが示された。



一方、素系の範囲を越えて後戻りを行うことは可能である。例えば、βよりも前の奇数番目のノードには後戻りができる。しかし、これを繰返すとβは最終的に灯台に到達し、それ以上前のノードに後戻りすることはできなくなる。また逆にαを越えた後、改めてβに後戻りすることも可能だ。しかし、これを繰返すとαは最終的に共有点に到達することになる。

以上の検証により、素系においては無限ループが発生しないことが確認できた。したがって、全体系でも無限ループは発生しない。これにより、到達点は必ず共有点に達することが示され、分断ルートを開通する解の存在が証明された。

12. まとめ

本稿では、地図を四色で塗り分ける確実な手順を示すことで、四色問題の論理的証明を与えた。

まずは、地図の平面グラフ化とそこへの補助ライン追加により、還元可能な三角ネット網に単純化できることを示した。グラフが単純化されたため、着色プロセスはノード接続とライン接続の2種類に集約され、同色ライン接続以外の着色は四色での塗分けが保証されることを示した。そして残された同色ライン接続を解消するために、「排他的な色の組合せで構成される2つの連鎖ルートは交差しない」という定理を導き、分断ルートの開通という手法を見出した。

続いて分断ルートの開通手順を提示した。その手順は、対象とする端点に直接接続されたノードを初期の分断ルート予定区間とし、スワップを利用して分断ルートを順次開通させていく手法である。本手順は後戻経路を検出した際、予定区間を補正していく内容も含んでいる。最後に、本手順が分断ルートの解を必ず得られることを示すために、後戻パターンの最小要素である素系と、それらの輻輳関係を検証することで、到達点が確実に共有点に向かっていくことを確認した。

以上の論旨展開により、計算機を用いることなく、あらゆる地図が四色で塗り分けられることが証明された。本稿が提案する手順は与えられた地図の場合分けを伴わず、コンピューターによる証明のように膨大なケースを準備する必要は無い。また、本稿が導き出した分断ルートに関する定理は、何故地図の塗分けは3色では足りず、5色では多すぎるのかという根本的な問いへの答えも提供する。「排他的な色の組合せで構成される2つの連鎖ルートは交差しない」という定理の内容は、独立な2色の連鎖ルート2本の関係を示すものであり、2色×2本、すなわち4色が必要十分な色数だからである。

Q. E. D