グラフ理論を用いた構造異性体の数え上げ

【研究の動機】

学校のAL活動でグラフ理論を学び、　　競技プログラミングを学ぶ途中 Pythonでグラフ理論の問題を解いた。このとき Pythonでグラフを扱って何かを調べることはうできないだろうかと思い、グラフの形状から構造式が思いついたので、グラフ理論で物質の構造異性体を全部数えるプログラムを組もうと考えた。　　今回はアルカンや有機物を対象にした。

【調べる方法】

①水素は最後に繋がってない部分につければいいので無視し、炭素と酸素の結合方法を調べる。 ②炭素と酸素を頂点と見立てたグラフを作る、このとき炭素と酸素の区別はつけないものとし、 水素、酸素、炭素の数から、水素と繋がっていない辺の数を導き、辺の繋ぎ方をすべて列挙する。③各頂点について、一番遠い頂点に至るまで　最低でも通る必要がある辺の数 (以降距離と呼ぶ)と一番遠い頂点の数、各頂点までの距離の度数を調べる。自己ループ、サイクルは考えない。　④それまでに出た結合方法と被っていなければ、炭素と酸素の区別をして、酸素を置く位置のすべての場合について③と同じことをする。それまでに出た結合と被っていなければ 1つ数える。

【実験1】 アルカンの構造異性体

アルカンはメタン、エタンといった「 C_nH_{2n+2}」の式で表される物質である。アルカンは全ての原子が単結合で繋がれているため、炭素を頂点とする単純グラフとみなせる。

結果

炭素の数	1	2	3	4	5	6	7	8	9
構造異性体の数	1	1	1	2	3	5	9	18	35

このプログラムでは N=9の場合を調べるのに 1時間を要したため、これ以上進めることができなかった。可能である辺の組み合わせを全て調べたため、調べるパターンが $(N(N+1)/2)C_{N-1}$通りになっていたため、組み合わせ爆発が起こったため、このように大幅に時間がかかったと考えられる。　　そこで最初に,ありえる次数の組から辺のスタート地点を決めて、ゴール地点を決めて、対応するように辺を足すようにすることで計算量を減らす。

【実験2】　プログラムの高速化

先述のようにスタート地点とゴール地点を定めてから辺を追加するようにコードを変えて実行してみたところ、実験 1では1時間かかっていた C=9の場合の実行時間が54秒までに短縮された。　　C=10の場合異性体は75個、実行時間が 5分9秒,C=11の場合異性体は159個、実行時間は約 34分であった。どちらも正しく出力されていた。

【実験3】　酸素を加える

アルカンだけだった実験対象をさらに広げ、有機物や、アルカンでない一部の炭素水素化合物についても調べた結果、次のような結果が得られた。

C,H,Oの数	構造異性体の数	実行時間
C2,H6,O1	2個	3秒
C3,H8,O1	3個	3秒
C4,H10,O1	7個	5秒
C5,H10	5個	4秒
C8,H10,O1	5個	11分21秒

C8H10Oの構造異性体の数が非常に少なく出たのはサイクルのあるグラフを考えていない、つまり環状構造を考慮していないからと考えられる。 C8H10Oの構造異性体はほとんどがベンゼン環を持っているため、ベンゼン環を持たないものだけが出力されたと考えられる。

環状構造を許して再び調べると、今度は存在しない環状構造をいくつも作り、結果出力は数百個にもなった。

【まとめ、展望】

扱う物質によっては計算量が膨大になってしまう、出力された構造異性体が実際に存在するかどうかを考慮していないなど、まだ問題点が十分にあると考えられる。今後は改良を重ねて、最終的には構造異性体を求められる計算ツールを作りたい。

【参考文献,コード】

今回の実験で使ったコード:https://colab.research.google.com/drive/1DFIi5DRO5l1Vjdqx2PSCO79eZPPPZ5Go?usp=sharing

参考にした記事:networkxの基本的な使い方
https://qiita.com/kzm4269/items/081ff2fdb8a6b0a6112f