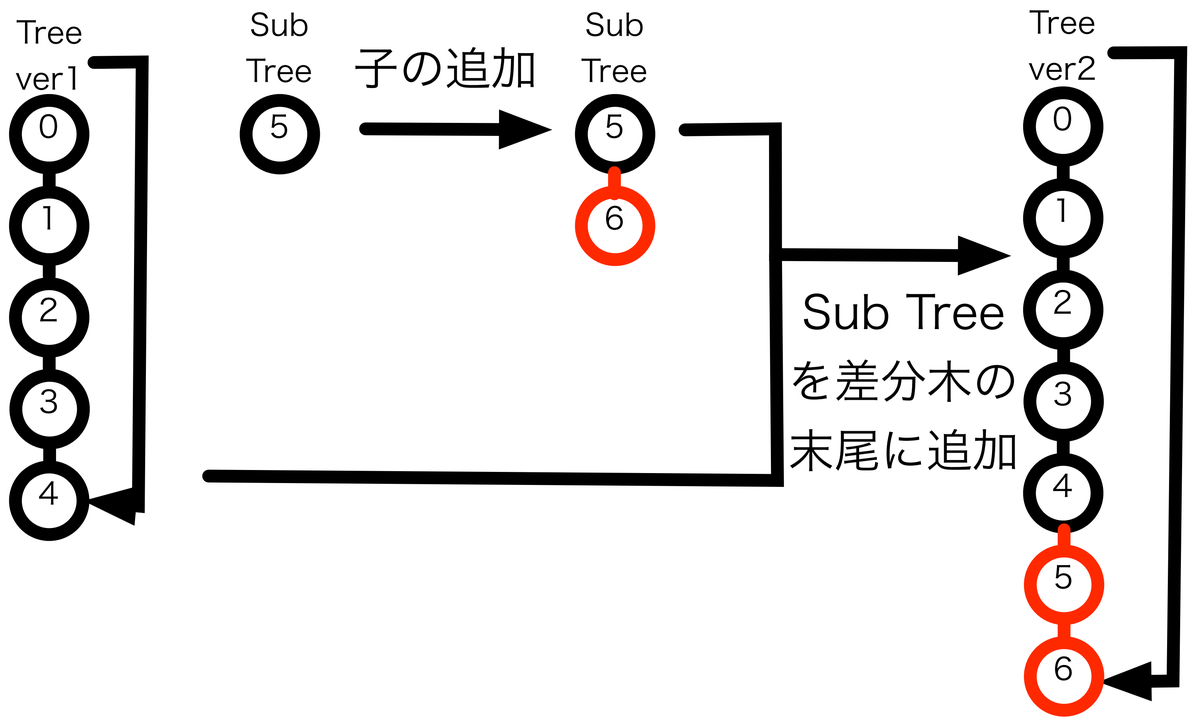
差分木の編集

・差分木へのノードの追加はAtomicに行う必要があるため、複数のノードを追加する際は複数回のトランザクションを行う必要があった
・差分木への複数個のノードの追加を1回のトランザクションで行うために、別途Sub Treeを構築しそれに対して破壊的に編集を加え、最後に差分木に追加する手法を提案する



Jungle上での巨大な木の扱い

・Jungleは木の編集時に木の複製を行うため、木の変更の手間は木構造の大きさにも依存している
・バランスの取れた木構造を構築することで、編集の手間をO(log n)にすることが可能であるが、ユーザーが全ての木のバランスを取ることは難しい
・そこで、木の生成時に指定した属性名でバランスを取り最適な木構造を構築する機能を持つ、Red Black Jungle Treeを実装した
・Red Black Jungle Treeは、任意の形の木構造を構築することはできない
・木の生成時指定した属性名と、それとペアの属性名を用いて任意のノードにO(logn)でアクセスすることができるため、Indexを構築する必要がない

NodePathの拡張

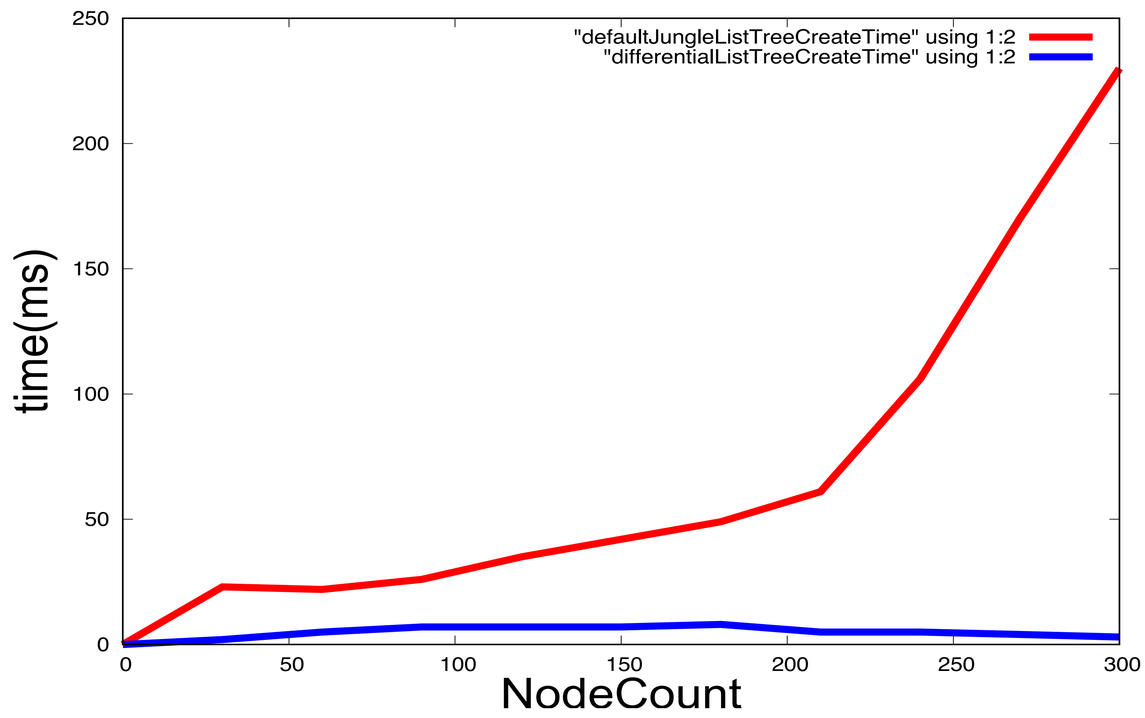
・Red Black Jungle Treeは、ノードを追加、削除するたびに木のバランスが行われ、木の形が変わり各ノードへのパスが変わってしまう
・その為、木に編集を加える際に対象のノードのPathを毎回調べる必要がある
・その問題を解決するためにノードを数値ではなく、ノードが持つ属性名と属性値の組で指定できるようにPathを改良した
・ノードの指定に使用する属性名は、木の生成時に指定した属性名を用いる必要がある
・これは、Red Black Jungle Treeが、生成時に指定した属性名でソートされているからである

性能評価

今回新しくJungleに追加した機能の測定を行う
測定は
・線形の木を構築する時間
・Red Black Jungle Treeの構築時間
の2つの項目を行う

正順線形木の構築時間の測定

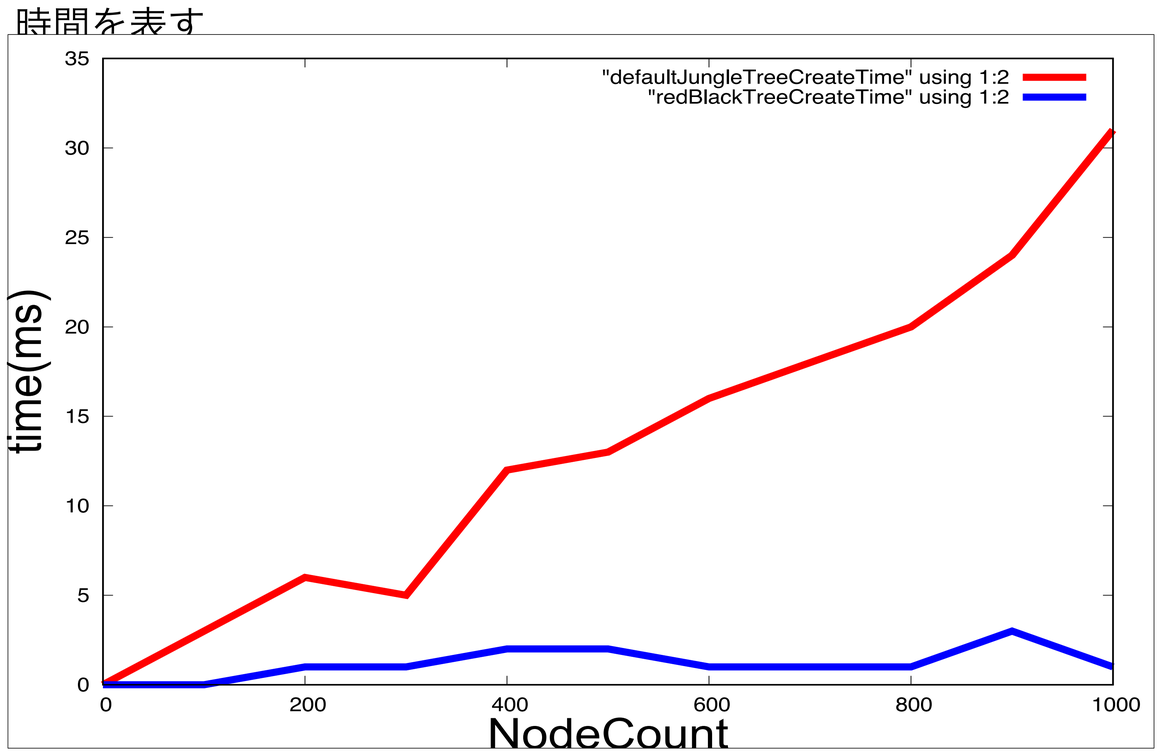
・測定は既存のJungle Treeと新しく実装した差分木で行う
・グラフのX軸は、構築した木のノード数、Y軸は木の構築にかかった時間を表す
・今回の測定ではIndexは構築していない



・差分木は構築する木のサイズが変わっても構築時間はあまり変らない
・一方既存のJungle Treeは、木の構築時間がn ^ 2で増えていっている
・これは、既存のJungle Treeがノードを追加するたびに全てのノードの複製を行ってるのに対し、差分木は複製を行っていないからである

Red Black Jungle Treeの性能評価

・測定は既存のJungle Treeと新しく実装したRed Black Jungle Treeで行う
・グラフのX軸は、構築した木のノード数、Y軸は木の構築にかかった時間を表す



・Jungleの既存の木に比べ、Red Black Jungle Treeの方が高速に木の構築を行えている
・これは、既存のJungle Treeが木の構築後、Indexを構築しているのに対し、Red Black Jungle TreeはIndexを構築していないためである

今後の課題

・Jungleは多くのメモリを使用するため、ある程度の単位で過去のデータを掃除する必要がある
・Jungleが十分なパフォーマンスを出すためには、データを最適化する必要があるが、最適な木構造はJungleを使用するアプリケーションによって違うため、Jungleの設計手法を確立させる必要がある