ポスター | 病院情報システム
ポスター13
病院情報システム
2019年11月24日(日) 10:00 〜 11:00 ポスター会場2 (国際展示場 展示ホール8)

[4-P2-2-06] ブロックチェーン技術を用いた単一医療機関向け診療記録システムに実装したデータ同期機能の検証

○関口 諄[1]、五味 悠一郎[2] （1. 日本大学理工学部応用情報工学科, 2. 日本大学理工学部）

キーワード：block chain, medical records system, single medical institution

ビッグデータ技術や人工知能の発展により，医療機関が保有する膨大な患者の医療データを適切な診断や新治療法発明などに役立てることが期待されている。膨大な医療データを管理し共有するためには，高いセキュリティ基準や，複雑なアクセス制御が可能なシステムが求められる。このようなシステムを中央集約型の集中管理式で構築する場合，その導入や運用コストと手間は膨大なものになる。しかし，分散型台帳システムのブロックチェーンを用いることで導入と手間を解決することができる．

提案されているブロックチェーンを用いた診療記録システムの多くは複数医療機関で利用する方式であり，医療機関同士で事前に合意形成する必要があるため，利用する対象を全国と設定すると難しい。ただし，地域や関連病院および単一医療機関といった形で対象の範囲を狭めると導入も簡単となる．

先行研究ではブロックチェーンを用いた文字格納プログラムの一秒間に格納できる情報量が単一医療機関で一秒間に発生する情報量を上回ることや，単一医療機関向けの診療記録システムにブロックチェーンを用いても問題ないことが分かった．しかし，システムの機能として，複数サーバでのシステム稼働中にいずれかのサーバが停止し復帰した場合のデータ同期機能が未実装であり，システムに同時投稿が行われた際の問題の検証が行われていなかった．

本研究では，複数サーバでのシステム稼働中にいずれかのサーバが停止し，停止状態から復旧した場合にデータを同期させるプログラムを実装し，問題なく動作することを確認した．また，システムに同時投稿した際の問題を検証し，その問題の解決策を検討した．

ブロックチェーン技術を用いた単一医療機関向け診療記録システムに実装したデータ同期機能の検証

関口　諄[*1]、五味悠一郎[*2]、
*1 日本大学理工学部応用情報工学科、*2 日本大学理工学部

Examination of data sync function that implemented in medical records system for single medical institution using block chain system

Jun Sekiguchi[*1] , Yuichiro Gomi[*2]
*1 Department of Computer Engineering, College of Science and Technology, Nihon University
*2 College of Science and Technology, Nihon University

Block chain technology is a technology that ensures authenticity because the recording is irreversible. In addition, readability and preservability are secured P2P type system. This research is an evaluation of medical records system for single medical institution using this technology. Clarifying problem of medical records system using block chain technology and point of difference between medical records system using block chain technology and medical records system using distribute database technology.

Keywords: block chain, medical records system, single medical institution.

1. 緒論

ビッグデータ技術や人工知能の発展により、医療機関が保有する膨大な患者の医療データを適切な診断や新治療法発明などに役立てることが期待されている。膨大な医療データを管理し、共有するためには、高いセキュリティ水準や複雑なアクセス制御が可能なシステムが求められる。このようなシステムを中央集約型の集中管理式で構築する場合、その導入と運用のコストと手間は莫大なものになる。そこで、分散型台帳システムのブロックチェーン技術を用いて導入や運用コストと手間を解決することができる[1)]。

ブロックチェーンとは、仮想通貨であるビットコインの根幹技術として生まれた分散型台帳システムを指す。記録の改ざん困難性などの特徴が注目され、医療分野への活用も進められている[2)]。

診療記録等の法的に保存義務のある記録を電子保持する場合、電子保存の三原則である真正性と見読性および保存性の確保が義務付けられている。ブロックチェーン技術では、電子署名による文書作成者の証明とマイニングを利用した記録の非可逆性から真正性が確保されており、P2P システムを用いることで、見読性と保存性も確保されている[3)]。

ブロックチェーン技術を用いた診療記録システムは、いくつか提案されている。複数医療機関で利用する方式では、医療機関同士で事前に合意形成を行う必要があり、利用する範囲を全国に設定すると導入が難しい[2)4)]。そこで、地域や関連医療機関といった形で対象の範囲を狭めると、導入も簡単になる。

我々がブロックチェーン技術を用いて構築したブログシステムのメディカルシステムαver.3 では、一時停止していたサーバが復帰した際に同期を行う同期機能の実装を試みたがプログラムの不備があったため、同期が正常に行われなかった。本研究ではメディカルαver.3 の同期機能を見直してメディカルαver.4 として検証を行った。

2. 開発目的

メディカルαver.3 を見直し、同期機能を正常に動作させることでブロックチェーンにおける同期システムの課題や利点を明らかにする。

3. システム概要

メディカルαver.4 は、診療記録システムとして最低限必要な入力機能と編集機能および削除機能を備えたブログシステムに、サーバが一時停止の状態から復帰した際にほかのサーバと同期を行う同期機能を実装したシステムである。

メディカルαver.4 は、1 台のパソコンと 3 台の Raspberry Pi でローカルネットワーク上に構築した。台の Raspberry Pi のうち 1 台を DHCP サーバと DNS サーバとして設定し、残りの 2 台の Raspberry Pi を web サーバとして設定した。web サーバとして設定した 2 台の Raspberry Pi の IP アドレスを 192.168. 10.1 と 192.168. 10.2 と設定し、192.168. 10.1 のホスト名を Pi1、192.168. 10.2 のホスト名を Pi2 とした。

3.1 システム構成

メディカルαver.4 は、以下のファイルとプログラムから構成されている。

1) index.php
   ブログシステムのメインプログラムで、このプログラムに同期機能を実装した。
2) blockchain.php
   ブロックチェーンを操作するためのプログラム。
3) get_prev_cnt.php
   要素数であるチェーンの長さを返すプログラム。
4) memory_temp.txt
   ブログシステムに書き込んだ文章を保存するファイル。
5) memory_chain.txt
   メディカルαver.4 に書き込まれた時や、書き込まれた内容が編集および削除されたときに、生成されたブロックを時系列順に追記していくファイル。

3.2 同期機能のアルゴリズム

メディカルαver.4 に実装した同期機能のアルゴリズムを図 1 に示す。自サーバは同期プログラムを実行するサーバのことであり、相手サーバは同期する相手のサーバのことである。

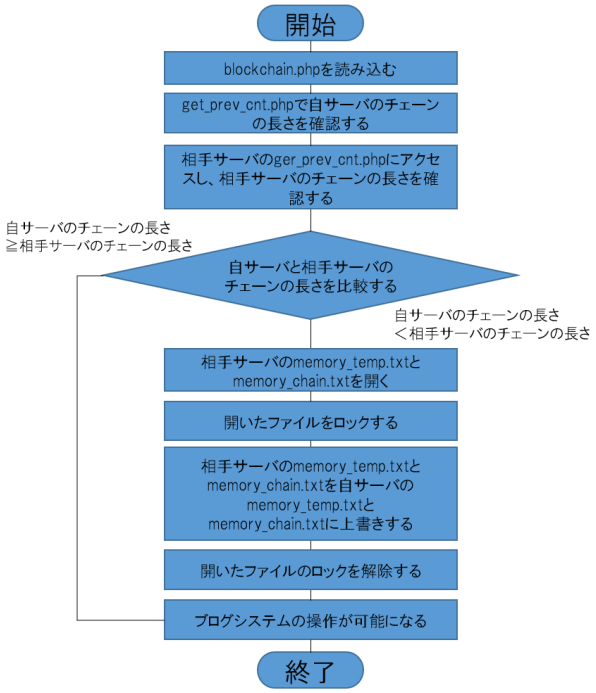

図 1　同期機能のアルゴリズム

3.3 システム要素技術

図 1 に示した処理内容を 1)~3)で説明する。

1) blockchain.php を読み込む
　メディカルαver.4 内のブロックチェーンを操作するためのプログラムである blockchain.php を読み込む
2) 開いたファイルをロックする
　自サーバが相手サーバの memory_temp.txt と memory_cahin.txt を開き、同期のための処理を行っているときは、アクセスしている自サーバと相手サーバのファイルにロックをかける必要がある。なぜなら、ファイルを開き同期処理を行っている最中に要素数の多いサーバからファイルに対して書き込みがあると、データが破損してしまうためである。
3) 相手サーバの memory_temp.txt と memory_cahin.txt を自サーバの memory_temp.txt と memory_cahin.txt に上書きする
　同期処理は、自サーバのチェーンが相手サーバのチェーンよりも短かった場合に相手サーバの memory_temp.txt と memory_cahin.txt を自サーバの memory_temp.txt と memory_cahin.txt へコピーする方法を用いた

3.4 メディカルαver.3 からの変更点

メディカルαver.3 からの変更点は 2 つあり、Pi1 上で稼働した場合を例として示す。Pi2 上で稼働した場合も変更点は同様である。

1 つ目は要素数を確認する方法を変更した。メディカルαver.3 では Pi1 が Pi2 の要素数を確認する方法として、Pi2 のサーバ上で稼働しているブログシステムの web ページ全体を読み込むことで要素数の確認を試みていた。そのため、要素数の確認がうまく行われず、Pi2 上で稼働しているブログシステムの web ページ全体が Pi1 上の画面に表示されていた。メディカルαver.4 では Pi1 と Pi2 が互いの要素数を確認する方法として、互いのサーバ上に記録されたチェーンの長さを要素数とし、チェーンの長さを数えることで要素数の確認を行った。

2 つ目は同期の方法を変更した。メディカルαver.3 では Pi1 と Pi2 の要素数を比較した後に要素数の差分を計算しようとした。しかし、要素数の確認を正常に行うことができず、要素数の差分を計算することはできなかった。メディカルαver.4 では同期を行うことを第一としたため、Pi1 と Pi2 の要素数を比較した後は差分を取らないことにした。要素数の少ないサーバが要素数の多いサーバの memory_temp.txt と memory_chain.txt を開き、自サーバの memory_temp.txt と memory_chain.txt へ上書きを行うように変更した。

4　システム評価

Pi1 と Pi2 でそれぞれブログシステムを稼動し、Pi1 の入力機能で「abc」と書き込みを 1 度行った結果、Pi1 に正常に書き込むことができた。その後、Pi2 上で稼働しているブログシステムを再読み込みすると、Pi1 で書きこんだ「abc」の文字とブロック番号とブロックが作成された時刻（書き込みが行われた時刻）およびハッシュ値が反映されており、書き込んだテキストだけでなくブロックチェーンに記録されている要素すべてが反映されていた。Pi2 でも同様の書き込みを行った結果、Pi1 に書き込みを行った結果と同じ結果を得ることができた。

次にキーボードとマウスの動作を行う汎用マクロソフトの「KeyToKey」を用いて連続書き込みテストを行った [5]。Pi1 と Pi2 を稼働させ、Pi1 に「abc」と書き込みを行った後に Pi2 上で稼働しているブログシステムの再読み込みをし、同期を行うという一連の動作を「KeyToKey」の記録機能を利用して記録した。その後、記録した一連の動作を「KeyToKey」の再生機能を利用して 4000 回再生を行い、4000 回の書き込み処理と同期処理を行った。その結果、4000 回の書き込みと同期処理は正常に行われた。しかし、書き込み回数と同期回数が増えるにつれて動作が遅くなったように感じたため Pi1 のアクセスログを確認した。すると、書き込み回数が約 1000 回を超える前は 1 回の書き込みから同期の終了までにかかる平均時間が約 3 秒であるのに対し、書き込み回数が 1000 回を超えると書き込みから同期までにかかる平均時間が約 7 秒となっていたため、動作が遅くなっていたことが確認できた。また、書き込み回数が約 1000 回でデータ容量が1MB を超えていた。Pi2 に「abc」と書き込みを行った後に Pi1 を再稼働させ同期を行った場合の動作も同様に記録し、4000 回再生した場合も書き込み処理と同期処理は正常に行われ、書き込み回数が 1000 回を超えたあたりで動作が遅くなることもアクセスログから確認することができた。

5. 考察

メディカルαver.4 を稼動した結果、正常に書き込みや同期を行えたことから ver.3 からの変更点である自分のブロックチェーンと相手のブロックチェーンの要素数を比較する際の要素や、要素数を確認する際の方法と同期を行う際の処理は正しかったと言える。

メディカルαver.4 の動作確認として 4000 回書き込み処理と同期処理を行った結果、正常に書き込みと同期処理が動作したが、書き込み回数が約 1000 回を超えるとデータ容量が 1MB を超えメディカルαver.4 の動作が重くなることが確認できた。本研究で構築したメディカルαver.4 の同期機能は、書き込まれたテキストデータと生成されたチェーンの長さを Pi1 と Pi2 の要素として比較を行い、チェーンの短いほうのサーバがチェーンの長いほうのサーバの memory_temp.txt と

memory_chain.txt を開き、チェーンの短いサーバの memory_temp.txt と memory_chain.txt へ上書きをすることで同期を行った。そのため、書き込み回数が多くなるほど同期を行うデータ容量が大きくなってしまいメディカルαver.4 の動作が遅くなったと考えられる。診療記録は 5 年間の保存が義務付けられており、同期を行うたびにすべての診療記録データを同期するのに膨大な時間がかかってしまう [6]。その対策として互いの要素数の比較した際、要素数の差分のみをコピーし追加することで、同期にかかる時間を削減することができると考える。そのほかに動作が重くなった理由として、メディカルαver.4 は画面上に書き込んだテキストを入力フォームごと表示していたことが考えられる。書き込まれたテキストのみを表示することができれば、入力フォームを表示することによる動作が遅くなることはなくなると考える。

6. 結論

本研究では、メディカルαver.3 のアルゴリズムやプログラムを見直して新たにメディカルαver.4 を構築した。メディカルαver.4 を稼動した結果、正常に同期が行われることが確認できたが、書き込まれた回数が増えるとデータ容量が増えてしまうため、メディカルαver.4 の動作が遅くなることが確認できた。その対策として、メディカルαver.3 で試みた要素数の差分のみを追加する方法で同期を行う必要がある。また、要素数の確認をチェーンの長さのみで確認を行っており、同期をする前のデータの担保ができていないため、確認する要素はチェーンの長さだけでなく、記録されているハッシュ値などを追加する必要がある。

実際に診療記録システムとして稼働した際に同期が行われる状況を想定すると、稼働していたサーバを新しいサーバと交換した時とサーバの台数が増えた時のような新しいサーバがネットワークに参加した場合と、一時停止してしまったサーバが再起動した場合が考えられる。この 2 つの場合の同期については検証が行えていないため、検証する必要がある。

本研究では 2 台の Raspberry Pi を web サーバとし、同期元のサーバの決定は同期元となる IP アドレスを指定して行っていた。しかし、実際に医療現場で稼動させる際にはたくさんのサーバで稼動することになるため、今後はサーバの台数を増やした際の同期機能について同期元となるサーバの決定方法などを検討する。

参考文献

1) 岩崎淳也, 酒井正夫. 生体医工学 55Annual「ブロックチェーン：医療機関間の安全で自律分散的なデータ連携に向けて」2017：227－227
2) 大下淳一. ブロックチェーンは医療にどう活用できるのか. 2018 [https://tech.nikkeibp.co.jp/dm/atcl/feature/15/050200094/011900011/?ST=health&P=3(cited 2019-March-26)]
3) 加嵜長門,篠原航.ブロックチェーンアプリケーション開発の教科書 [https://guide.blockchain.z.com/ja/docs/oss/medical-record/ (cited 2019-April-12)]
4) GMO ブロックチェーン オープンソース提供プロジェクト 2017 [http://www.gmo.jp/mews/article/?id=5736(cited 2019-April-12)]
5) KeyToKey 開発ブログ [keytokey-dev.net/(cited 2019-August-31)]
6) 医療情報システムの安全管理に関するガイドライン 第5版 [https://www.mhlw.go.jp.file/05-Shingikai-12601000-0000166260.pdf(cited 2019-April-29)