2J-3

交通シミュレータの自動検証機能と類似性評価について

渡部　純也† 中村　俊一郎† 大澤　洋† 宮西　洋太郎‡
日本工業大学† 宮城大学‡

1.はじめに

都市及びその近郊における交通渋滞は深刻な状況であり、その解析・改善に向けて我々は交通シミュレータ NITTS(NIT Traffic Simulator)の開発を進めている。世の中には数多くの交通シミュレータが存在するが、我々は他のシミュレータには無い独自機能の実現を目指している。を及ぼしていない事を評価する検証機能である。自動検証機能とは、プログラムを変更した場合に、車１台１台の動きが変更前と変更後で変わらない事をチェックするプログラムであり、意図した変更が他の部分に悪影響を及ぼしていない事を自動的に確認出来る事を狙ったものである。

2.自動検証プログラムの開発目的

我々が開発を進めている交通シミュレータは、様々な新機能やプログラム改修が加えられ、大きなソフトになってきた。その為、ある追加プログラム・修正プログラムによって他の部分に悪影響が及ぶ事もしばしば起こるようになってきた。この為１つの不具合が発生し、その不具合を解消した場合、それ以外の場所に新たな不具合が発生していない事を確認する事が必須である。その為人手により他の全試験を流して再び確認する訳であるが、これは多大な時間と労力を費やすものであった。そこでその負担を軽減し、効率化を図る目的で”自動検証プログラム”の開発を行った。これは図 2 に示すように変更前のプログラムで各試験の走行データを記録し、それを改修後のプログラムのシミュレーション実行時と比較する事により、この改修が他の部分に悪影響を及ぼしていない事を確認するものである。なお、このためには多数の試験を流さなけ流さなければならないが、その為の複数試験連続実行機能を付与した。

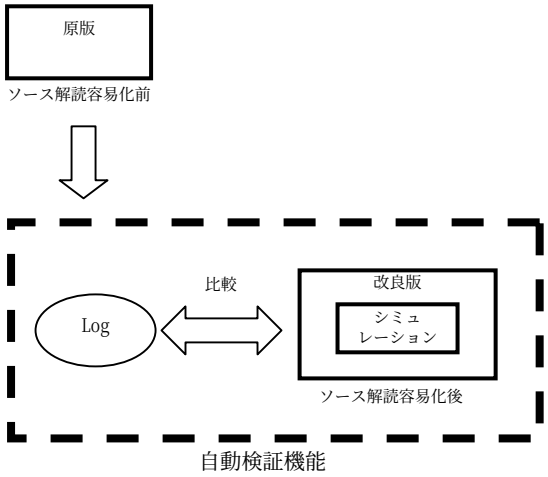


図 2　自動検証機能仕様例

3.自動検証機能動作

図 3-1 は従来のメニューフォームであるが、ここから図 4-2 自動検証フォームを呼び出し自動検証を実行する。自動検証は、大きく二つの機能に分かれる。第 1 は、データ書き込み機能である。この機能は、自動検証フォームからメタファイルを選択し、ファイル名を付け実行時間入力フォームで試験時間を入力すると、シミュレーションが実行され、走行データを log ファイルとして保存する。ファイルには車番号、道路番号、車線、変位、速度がそれぞれ 5 秒毎に書き込まれる。第 2 の機能は、検証機能である。データ書き込み機能によって保存された log ファイルと自動検証時の走行データを比較する。この機能により、試験の不具合、走行データのエラー等が発見される。

Automatic verification function and similarity evaluation function of a traffic simulator
†Junya Watabe,Shunichiro Nakamura,Hiroshi Ohsawa・Nippon Institute of Technology
‡Yohtaro Miyanishi・Myagi University



図 3-1　NITTS メニューフォーム



図 3-2　自動検証フォーム

4.ソース解読容易化への適用

我々の開発している交通シミュレータではプログラムソースコードのみが羅列されているものや大きなネスト構造が幾重にも重なったものが多数存在し解読が困難であった。そこで、プログラムソースコードの解読容易化作業を実施した。対象は動車系プログラム部分とし、全 40 サブルーチンに渡った。この結果、複雑で難解な動車系アルゴリズムの理解容易性が大いに高められた。このような人海戦術による人手作業にはミスの混入がつきものであり、その為自動検証機能が大いに役立った。ソース解読容易化ではソースコードのロジックを変更するわけではなく、ソースのコメントやソース内容にコメントで番号を付与しただけなので、容易化前と容易化後の走行データは一致するはずである。自動検証により 91 件の試験を連続実行させると実行時間は実時間シミュレーションした場合 68 分 15 秒の所、早送り機能を用いて、4 分 33 秒で完了する。この結果 NG となるので“データ不一致犯人割り出し試験”を行う。この方法は図 4 に示すように各自担当した容易化後サブルーチンを１つだけを容易化前のプログラムに組み込んだものを作成し、これに対して上記と同様に自動検証プログラムを走行させる。

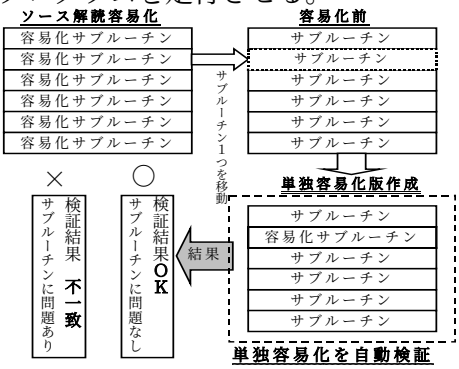


図 4　犯人割り出し試験方法

5.プログラム改修への適用

次に自動検証プログラムのプログラム改修への適用を試みた。これはロジック変更を伴う変更である為、原理的には自動検証プログラムの正常動作は保証されない。その結果を表 5 に示す。

表 5　2.7S 版と 2.75S 版の検証結果

	全サブルーチン	改修サブルーチン	OK試験数	NG試験数
原版	287	0	63	0
改修版	287	8	55	8

NITTS2.7S 版から 2.75S への改修において、全 7 件の改修が加えられ、287 件中９件のサブルーチン変更が加えられた。表 5 はこの改修前後のプログラムに対して自動検証プログラムを 63 試験連続実行させた結果である。63 試験中８試験で NG となったが、逆に言えば残りの 55 試験では全車の動きがピッタリ一致したということであり、一定の示唆を与えてくれる。

6.類似一致方式の検討

現在の自動検証は完全一致方式でデータ比較を行っており、類似性を判断する機能はない。自動検証機能をプログラム改修に適用出来るようにする為に”類似一致方式”の確立を模索中である。これはハッキリとした答えが出るものではないが、複数の試験結果を多面的に統計処理して、例えば類似度何％というような値が出せないかというようなものである。あるいは改修の種類に応じて何らかの傾向が見えてこないかというようなものである。その為の基礎データとして例えば表 6 に示すような車ごとの一致不一致のマトリックスを作成して、この解析を行う事等を検討中である。

表 6　類似性を使用した試験評価

試験No.	車1	車2	車3	車4	車5	車6	車7	車8	……	一致or不一致	不一致率
1	0	1	1	0	0	0	0	1	……	×	30%
2	1	1	1	0	0	1	1	0	……	×	50%
3	0	0	0	0	0	0	0	0	……	○	0%
4	1	1	1	1	1	0	0	1	……	×	80%
・	・	・	・	・	・	・	・	・	……	・	・
・	・	・	・	・	・	・	・	・	……	・	・
・	・	・	・	・	・	・	・	・	……	・	・
・	・	・	・	・	・	・	・	・	……	・	・

7.終わりに

交通シミュレーションの自動検証を開発し、プログラム開発体制に組み込んだ結果、ソース解読容易化のプログラム変更に対してはきわめて有効に動作している。これをプログラム改修に対しても有効に使えるよう、今後さらに改良を図っていく所存である。