รายงานวิจัยฉบับสมบูรณ์

โครงการ แพลทฟอร์มการคำนวณสำหรับการค้นหาเบสซ้ำในจีโนม และการค้นพบฟังก์ชันทางชีววิทยา

A Computing Platform for Finding Tandem Repeats in the Whole Genomes and a Discovery of their Biological Functions

โดย รองศาสตราจารย์ ดร. ชัชวิทย์ อาภรณ์เทวัญ และคณะ

ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์

คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

มิถุนายน 2561

สัญญาเลขที่ RSA5980060

รายงานวิจัยฉบับสมบูรณ์

โครงการแพลทฟอร์มการคำนวณสำหรับการค้นหาเบสซ้ำในจีโนม และการค้นพบฟังก์ชันทางชีววิทยา

A Computing Platform for Finding Tandem Repeats in the Whole Genomes and a Discovery of their Biological Functions

รองศาสตราจารย์ ดร. ชัชวิทย์ อาภรณ์เทวัญ

ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์

คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

สนับสนุนโดย สำนักงานกองทุนสนับสนุนการวิจัย และจุฬาลงกรณ์มหาวิทยาลัย

(ความเห็นในรายงานนี้เป็นของผู้วิจัย สกว. ไม่จำเป็นต้องเห็นด้วยเสมอไป)

บทคัดย่อ

รหัสโครงการ : RSA5980060

ชื่อโครงการ : แพลทฟอร์มการคำนวณสำหรับการค้นหาเบสซ้ำในจีโนม และการค้นพบฟังก์ชันทางชีววิทยา

ชื่อนักวิจัย และสถาบัน : รองศาสตราจารย์ ดร.ชัชวิทย์ อาภรณ์เทวัญ
ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์
จุฬาลงกรณ์มหาวิทยาลัย

อีเมล์ : Chatchawit.A@chula.ac.th

ระยะเวลาโครงการ : 16 มิ.ย. 2559 ถึง 15 มิ.ย. 2561

บทคัดย่อ :

เราได้ดำเนินการศึกษาโรคอัลไซเมอร์โดยใช้วิธีการ genome-wide association study (GWAS) โดยปกติ GWAS จะใช้มาร์คเกอร์เป็น single nucleotide polymorphisms (SNPs) แต่เราใช้มาร์คเกอร์เป็นไมโครแซทเทลไลท์จากทั้งจีโนมของมนุษย์ ความแตกต่างของไมโครแซทเทลไลท์ได้มากจากข้อมูล whole genome sequencing ของผู้ป่วยอัลไซเมอร์ 128 คน และคนปกติ 267 คน การวิเคราะห์ความแตกต่างของไมโครแซทเทลไลท์นำไปสู่การค้นพบยีนใหม่ที่ยังไม่เคยพบมาก่อนในการวิเคราะห์ด้วย SNP เราจำแนก 70 ตำแหน่งของไมโครแซทเทลไลท์ที่มีนัยยะสำคัญทางสถิติในยีน AKIRIN2, TOMM40, MCU, EML6 และยีนอื่น ๆ ยีนเหล่านี้มีนัยยะที่สัมพันธ์กับการเกิดโรคอัลไซเมอร์อย่างมาก และอาจจะเป็นเป้าหมายใหม่สำหรับการรักษาโรคนี้ แม้ว่าเรายังขาดความเข้ากลไกทางชีววิทยา แต่ไมโครแซทเทลไลท์ก็มีประโยชน์ในปัจจุบันสำหรับการทำนายการเกิดโรคอัลไซเมอร์และการทำนายอายุที่จะเป็นโรคอัลไซเมอร์

คำหลัก : ไมโครแซทเทลไลท์, โรคอัลไซเมอร์

Abstract

Project Code : RSA5980060

Project Title : A Computing Platform for Finding Tandem Repeats in the Whole Genomes and a Discovery of their Biological Functions

Investigator : Associate Prof. Chatchawit Aporntewan
Department of Mathematics and Computer Science,
Faculty of Science, Chulalongkorn University

E-mail Address : Chatchawit.A@chula.ac.th

Project Period : July 16, 2016 – July 15, 2018

Abstract :

We performed a genome-wide association study (GWAS) of Alzheimer's disease (AD). In contrast to conventional markers like single nucleotide polymorphisms (SNPs), we analyzed microsatellite markers in the whole human genome. Microsatellite variants were called from the whole genome sequencing data of 128 AD patients and 267 normal subjects. The analysis of microsatellite markers lead to the discovery of novel genes that had never been identified by means of SNP analysis. We have identified 70 statistically significant microsatellite loci in the AKIRIN2, TOMM40, MCU, EML6, and other genes. Those genes were substantially relevant to the pathogenesis of AD, and could become a new target for medication. Despite the lack of complete understanding in the biological mechanism, the microsatellite markers were immediately useful for predicting the chance of developing AD and the age of disease onset.

Keywords : Microsatellites, Alzheimer's disease

Executive Summary

Initially, the research project was broadly titled "A Computing Platform for Finding Tandem Repeats in the Whole Genomes and a Discovery of their Biological Functions." Later, we focused on a specific class of tandem repeats called "microsatellites" and their biological functions in Alzheimer's disease. We have built a computational platform to identify microsatellite markers in the whole human genome. However, we planned to publish the computational platform after the study of Alzheimer's disease and ther neurological diseases. At the time of writing this report, the study of Alzheimer's disease is nearly complete. We are writing the manuscript and preparing for submission to the Genome Research journal (IF = 11.351 in 2015). A tentative title of the manuscript would be "genome-wide scan of microsatellite markers unravels novel genes associated with Alzheimer's disease"

The published research paper namely "Indexing Simple Graphs by Means of the Resistance Distance" (Appendix 1) was a polynomial-time solution to the graph isomorphism problem. The idea was developed during the annual TRF meeting. The algorithm was correct for small graphs, but we could not prove the correctness for larger graphs. Although this research article did not overlap with the aim and scope of this grant, we had acknowledged TRF for kind support.

Table of Content

1. Introduction

"Alzheimer's disease (AD) is a progressive and fatal neurodegenerative disorder manifested by cognitive and memory deterioration, progressive impairment of activities of daily living, and a variety of neuropsychiatric symptoms and behavioral disturbances" [1]. AD symptoms in chronological order include short-term memory loss, loss of some motor skills and languages, long-term memory loss, disoriented, bedridden, and death due to infection such as pneumonia.

The pathogenesis of AD is characterized by the accumulation of amyloid plaques and neurofibrillary tangles in brain cortex [2]. Amyloid precursor protein (APP) locates at the membrane of brain neurons. One end of an APP molecule is in the cell, and the other end is outside the cell. It is believed that APP helps a neuron grow and repair itself after an injury. In a normal recycle process, an APP molecule gets chopped into three soluble peptides by an enzyme namely alpha secretase and gamma secretase. If alpha secretase is replaced with beta secretase, a leftover peptide is not soluble and creates a monomer call amyloid beta. These monomers bond together outside neurons and form beta amyloid plaques. The plaques between neurons disrupt cell-to-cell signaling. Beside, these plaques can trigger an immune response and induce inflammation which may damage surrounding neurons. As opposed to the beta-amyloid plaques, neurofibrillary tangles occur inside neurons. In a neuron, cytoskeleton is partly made of microtubules which transport nutrients inside the cell. A protein called tau supports the microtubules from breaking apart. It is hypothesized that beta amyloid plaques activate kinase enzyme which transfers phosphate groups to tau protein. Subsequently, tau molecules detach from microtubules, clump up, and form a neurofibrillary tangle which leads to apoptosis and cell death. The amyloid plaque and neurofibrillary tangle are illustrated in Figure 1.

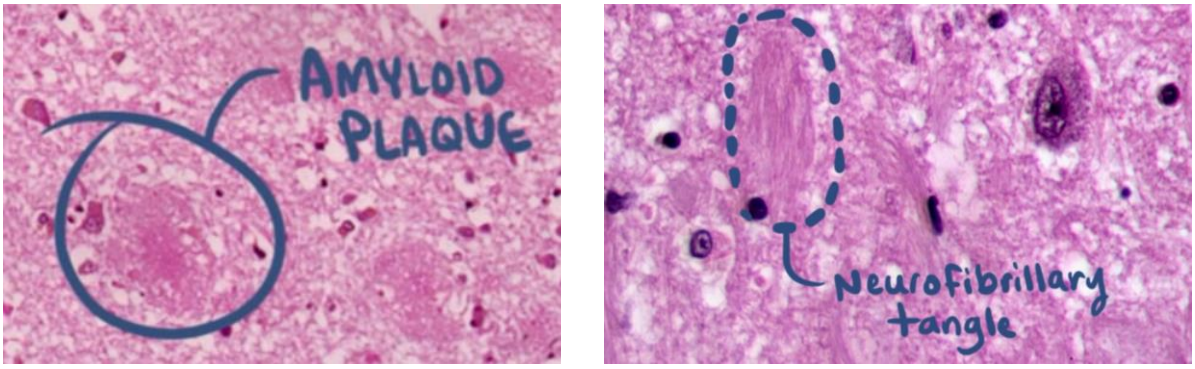


Figure 1. Amyloid plaques and neurofibrillary tangles.

In 2015, 46.8 million people are living with dementia (approximately 0.65% of the worldwide population). It is expected to be 74.7 million and 131.5 million in 2030 and 2050, respectively [3]. In Thailand, there were 600,000 AD patients in 2016 (approximately 0.91% of Thai population). It is estimated that the number of AD patients in Thailand will reach one million in 2029. There are several approaches for medications such as anti-amyloid therapies, neuro-protection, antioxidants, etc. However, there are no medications that clearly and definitively halt the progression of AD. As a result, AD is one of the most devastating diseases worldwide.

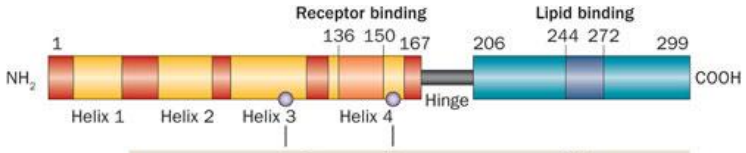
Genetic factors play a crucial role in developing AD [4]. People with some genetic variants are more susceptible to AD than the others. There are two major types of Alzheimer's disease: Early-Onset Alzheimer's Disease (EOAD) and Late-Onset Alzheimer's Disease (LOAD). EOAD or sporadic AD occurs between a person's 30s to mid-60s and accounts only 10% of all AD patients. At least three

causative genes (APP, PSEN1, PSEN2) have been identified. A genetic mutation in APP gene can alter the normal property of amyloid precursor protein. PSEN1 and PSEN2 genes encode proteins that are subunits of gamma secretase. A mutation on PSEN1 or PSEN2 genes can change the cleavage position on APP molecules. All these mutations lead to amyloid plaques. The genetic inheritance of these mutated genes is known as familial Alzheimer's disease (FAD). A child who carries a genetic mutation for FAD is highly susceptible to AD and early age of disease onset.

In contrast to EOAD, about 90% of AD patients are LOAD which occurs in a person's 60s and later [4]. From now on, AD means late-onset AD if not specified anything else. The genesis of AD is not yet completely understood. However, the risk of a person to develop AD seems to be affected by a combination of genetic factors, environmental factor, and life style. At present, no specific genes that directly cause AD have been found, but a genetic risk factor has been identified. A genetic variant in apolipoprotein E (APOE) gene on chromosome 19 substantially increases the risk of developing AD. There are 3 different alleles of APOE gene (rs429358 and rs7412).

- APO-ε2 is relatively rare as compared to APO-ε3 and APO-ε4. A person with this allele does not develop AD or develop later in life than a person with APO-ε4 allele. Thus, APO-ε2 is protective allele by reducing the risk and prolonging the onset of AD.
- APO-ε3 is the most common allele. It is believed that APO-ε3 does not decrease or increase the risk of developing AD.
- APO-ε4 increases the risk for AD and accelerates the age of disease onset. A number of APO-ε4 alleles in a person is proportional to the risk and the early onset of AD.

Apolipoprotein E helps break down beta-amyloid, but the APO-ε4 seems to be less effective than the other alleles. Therefore, the risk of AD increases with the number of APO-ε4 alleles. The amino acid difference of Apo-ε2, Apo-ε3, and Apo-ε4 proteins is shown in Figure 2.



	Isoform-specific amino acid difference		Allele frequency (%)	
	112	158	General	AD
Apo-E2	Cys	Cys	8.4	3.9
Apo-E3	Cys	Arg	77.9	59.4
Apo-E4	Arg	Arg	13.7	36.7

Figure 2. The amino acid difference of Apo-E2, Apo-E3, and Apo-E4.

Although APOE gene is an indisputable hallmark of AD, there are other potential genes such as SORL1, A2M, GST01, GST02, and GAB2 [5]. Gene association studies and genome-wide association studies have shown a number of genes that are statistically associated with the risk of AD [6]. A database was created to systematically manage a large number of conflicting reports. As of June 2018, the top ten genes on www.alzgene.org are APOE, BIN1, CLU, ABCA7, CR1, PICALM, MS4A6A, CD33, MS4A4E, CD2AP, respectively.

Single-nucleotide polymorphisms (SNPs) are commonly used as genetic markers for studying AD. In contrast, microsatellites or tandem repeats of 1-6 bp are abundant in the human genome and usually show high levels of length polymorphism due to DNA replication [7]. For instance, ATATATATAT is called a di-nucleotide repeat because the repeat unit (AT) is repeated five times. A form of mutation called replication slippage or slipped-strand mispairing can add or remove multiple units simultaneously. Microsatellite instability can cause many neurological diseases [8,9]. However, microsatellites have been rarely reported in the etiology of AD [10,11,12,13,14]. Recent findings show that polymorphic mononucleotide T-repeat in an intron of the TOMM40 gene is associated with AD risk [15,16]. Moreover, the length of T-repeat can predict the age of AD onset. The protein encoded by this gene is localized in the outer membrane of the mitochondria (MT). Mitochondrial dysfunction and oxidative stress are a common property of neurodegenerative diseases where MT struggle to provide sufficient energy for the cell [17]. Nevertheless, the biological relevance between TOMM40 and AD remains largely unknown.

The T-repeat (rs10524523) in the TOMM40 gene was discovered by deep sequencing a region of linkage disequilibrium that encompasses three genes (APOE, TOMM40, and APOC1) on chromosome 19. In this paper, we have investigated microsatellite loci in the whole human genome using whole genome sequencing (WGS) data. Our analysis shows that 70 polymorphic microsatellites pass the statistical threshold for genome-wide significance. Surprisingly, the T-repeat in the TOMM40 gene is in the list as well as other microsatellites in other genes that play a crucial role in neurological functions.

2. Materials and Methods

2.1 ADNI database

In 2012 – 2013, the whole genome sequencing (WGS) of 818 participants was conducted by Alzheimer's Disease Neuroimaging Initiative (ADNI) [18,19]. All subjects comprise of 128 AD, 415 mild cognitive impairment (MCI), 267 normal control subjects, and 8 of uncertain diagnosis. Only AD and control subjects were included in the analysis. With these subjects, 2 subjects were discarded due to low quality. All microsatellite loci were extracted from the variant call format (VCF) file in which the variants were called by ADNI using Broad's Best Practices. The VCF file also provided SNP data for our analysis. In addition, ADNI database also included the T-repeat alleles (rs10524523) of which the genotyping was performed by Polymorphic DNA Technologies using PolyT assays.

2.2 Mrep

All allelic sequences in the VCF file were input to the computer program called Mrep [20]. Only mono-, di-, tri-, tetra, penta-, and hexa-mononucleotide repeats were in our scope. Mrep detected exact repeats by the following parameters.

mrep -minp 1 -maxp 6 -exp 2.0 -xmloutput <filename> -s <sequence>

Approximate repeats allowed mismatches, insertions, and deletions. Mrep detected approximate repeats by the following parameters.

mrep -minp 1 -maxp 6 -res 3 -exp 2.0 -xmloutput <filename> -s <sequence>

Finally, all microsatellites loci in the whole human genome were identified.

2.3 Statistical Methods

Every genetic marker was equipped with a p-value which showed its statistical association with AD. There were only two types of genetic markers in our analysis. Firstly, the p-value of a single nucleotide polymorphism (SNP) was calculated using Fisher's Exact test [21]. Secondly, the p-value of a microsatellite was calculated using Anderson-Darling test which compared the distribution of microsatellite length (bp) between AD and normal subjects [22]. The null hypothesis is no difference between the two distributions. Bonferroni method was applied throughout the paper to correct the multiple hypothesis testing

2.4 Mayo Pilot RNAseq Dataset

RNAseq-based whole transcriptome data of brain samples were publicly available in [23]. The dataset consisted of two different brain regions, namely cerebellum (CER) and temporal cortex (TCX) brain regions. CER samples were categorized as 86 Alzheimer's disease (AD), 84 progressive supranuclear palsy (PSP), 28 pathologic aging, and 80 controls without neurodegenerative diagnoses. TCX samples were categorized as 84 AD, 84 PSP, 30 pathologic aging, and 80 controls. A differential expression analysis at gene level was conducted and provided with the dataset.

2.5 Ballgown

We performed a transcription-level expression analysis of the Mayo Pilot RNAseq Dataset using a software suite tool namely Ballgown [24]. We strictly followed every step as described in the paper except the read alignment which had been done at Mayo Clinic using the SNAPR software [25].

3. Results

3.1 The accuracy of microsatellite sequences from WGS data

First of all, we validated the accuracy of whole genome sequencing on microsatellite sequences. This can be done easily by compared the length of T-repeats (rs10524523) in the TOMM40 gene. The repeat length were independently measured by using two different methods, the PolyT assays and the whole genome sequencing (WGS). Only 156 subjects were measured by the two methods. A comparison of repeat lengths are shown in Figure 3.

Assuming that the repeat lengths from Poly-T assays are correct, the microsatellites called from WGS are a fair approximation. Although WGS do not perfectly replicate the repeat lengths, WGS is able to separate short (<25 bp) and long alleles (≥ 25bp) at 77.88% accuracy.

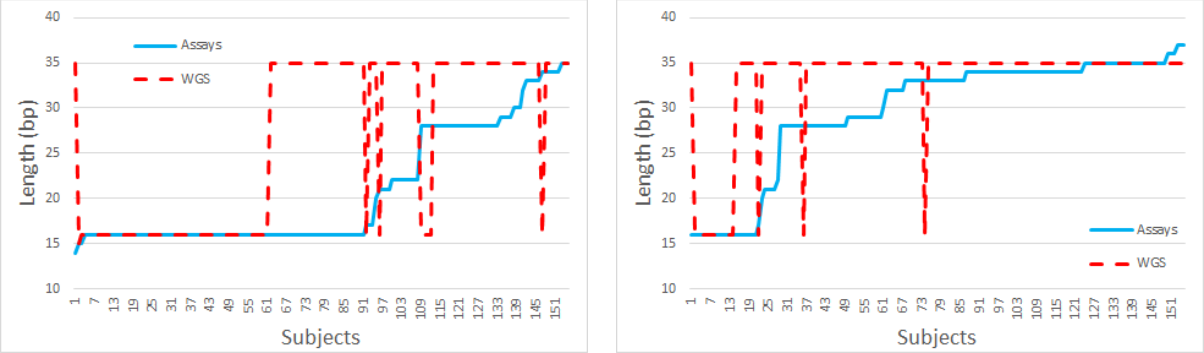
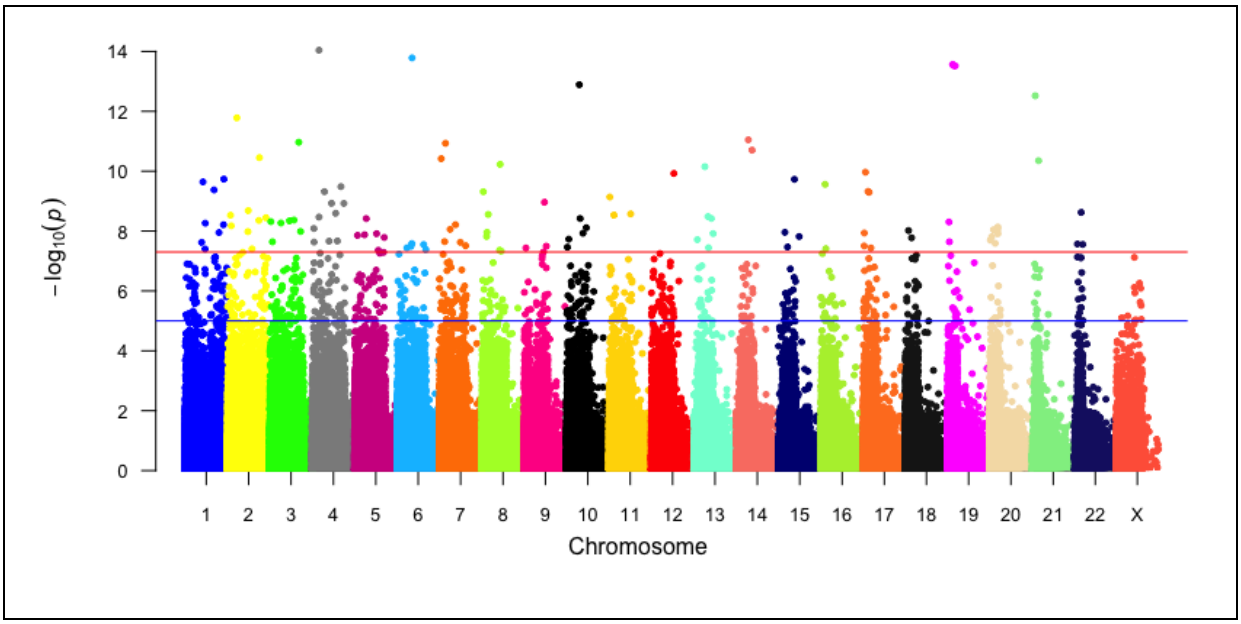


Figure 3. A comparison of repeat lengths called from assays and WGS. A subject has a pair of T-repeat alleles. One is the minimum-length allele (left), and the other one is the maximum-length allele (right). The correlation coefficients are 0.50 (left) and 0.62 (right).

3.2 There are several microsatellite loci that might be associated with AD

The result of genome-wide scan of microsatellites is shown in the Manhattan plot below. A total of 70 microsatellite loci pass the Anderson-Darling test (ADtest) and Bonferroni correction ($p \leq 0.05$). All significant microsatellite loci are found in the introns of protein-coding genes. Most microsatellites are mono- and di-nucleotide repeats, while there are a few of tri-, tetra-, and penta-nucleotide repeats. The signs of logistic regression coefficients are either positive or negative equally. The list of 70 statistically significant genes are summarized using DAVID Bioinformatics Resource 6.8 [26,27]. We found that 50 genes (71.4%) produce alternative splicing Protein for which at least two isoforms exist due to distinct pre-mRNA splicing events. 24 genes (34.3%) encode proteins that are found in the cytoplasm, the content of a cell within the plasma membrane and, in eukaryotic cells, surrounding the nucleus. This three-dimensional, jelly-like lattice interconnects and supports the other solid structures. The cytosol (the soluble portion of the cytoplasm outside the organelles) is mostly composed of water and many low molecular weight compounds. In eukaryotes, the cytoplasm also contains a network of cytoplasmic filaments (cytoskeleton). 34 genes (48.6%) are highly expressed in brain tissues.



Chr	Position	Repeat	Logit Coef	Sign	Logit P	ADtest P	Bonferroni Correction	Gene	Loc	Call Rate
6	88406417	(A)n	2.33	+	1.64E-04	1.64E-14	7.78E-09	AKIRIN2	intron	23.35
19	45403048	(T)n	0.05	+	5.36E-07	3.04E-14	1.44E-08	TOMM40	intron	62.63
10	74540269	(GT)n	0.18	+	5.45E-03	1.29E-13	6.12E-08	MCU	intron	37.58
2	55135058	(AGAT)n	-0.27	-	1.54E-05	1.65E-12	7.83E-07	EML6	intron	66.03
14	93265617	(T)n	-1.42	-	2.69E-02	1.98E-11	9.39E-06	GOLGA5	intron	58.81
7	10662405	(AAGA)n|(AAGG)n	-0.17	-	7.25E-06	3.82E-11	1.81E-05	MGC4859	intron	67.3
8	107417369	(AT)n	0.06	+	2.03E-05	5.89E-11	2.79E-05	OXR1	intron	72.4
17	12462174	(T)n	-0.51	-	2.92E-03	1.08E-10	5.12E-05	LINC00670	intron	58.39
16	24659595	(T)n	-0.19	-	4.89E-04	2.75E-10	1.30E-04	TNRC6A	intron	26.75
9	118994045	(T)n	-0.41	-	1.30E-02	1.09E-09	5.17E-04	PAPPA	intron	8.28
4	113350995	(A)n	-0.7	-	4.00E-03	1.18E-09	5.60E-04	ALPK1	intron	34.82
4	185987071	(A)n	-0.05	-	5.65E-01	1.19E-09	5.64E-04	LINC02436	intron	40.76
2	122275434	(T)n	-0.08	-	2.26E-03	2.07E-09	9.82E-04	CLASP1	intron	59.45
22	40477311	(T)n	-0.8	-	3.04E-04	2.38E-09	1.13E-03	TNRC6B	intron	45.65
11	127204840	(GT)n|(GC)n	0.27	+	1.10E-03	2.67E-09	1.27E-03	LOC101929497	intron	52.02
13	102813966	(AAG)n	0.21	+	7.88E-05	3.78E-09	1.79E-03	FGF14	intron	21.02
3	142047545	(TA)n	0.07	+	9.69E-05	4.28E-09	2.03E-03	XRN1	intron	64.12
3	115883448	(AT)n	-0.7	-	5.39E-05	4.55E-09	2.16E-03	LSAMP	intron	56.69
19	8480629	(A)n	0.11	+	3.73E-02	4.97E-09	2.36E-03	MARCH2	intron	27.39
7	94565963	(GT)n	-0.43	-	1.14E-04	6.12E-09	2.90E-03	PPP1R9A	intron	91.72
1	225838550	(A)n	-0.71	-	5.71E-03	6.16E-09	2.92E-03	ENAH	intron	29.72
20	48042207	(TA)n	-0.11	-	2.92E-04	7.33E-09	3.48E-03	KCNB1	intron	84.5
18	18658302	(AC)n	-0.7	-	2.80E-05	9.61E-09	4.56E-03	ROCK1	intron	85.35
3	180868166	(TA)n*	-0.09	-	1.68E-04	1.03E-08	4.89E-03	SOX2-OT	intron	92.99
2	120317621	(T)n	-1.01	-	4.42E-04	1.04E-08	4.93E-03	CFAP221	intron	24.2
8	30000863	(A)n	-0.11	-	3.09E-04	1.10E-08	5.22E-03	MBOAT4	intron	98.94
15	37288175	(AT)n	-0.05	-	1.26E-04	1.10E-08	5.22E-03	MEIS2	intron	37.58
10	96334855	(T)n	0.27	+	1.53E-01	1.18E-08	5.60E-03	HELLS	intron	78.34
20	48046980	(CATC)n	-0.31	-	2.78E-03	1.22E-08	5.79E-03	KCNB1	intron	99.15
5	128928070	(A)n	1.62	+	3.70E-04	1.22E-08	5.79E-03	ADAMTS19	intron	16.99
5	58539274	(T)n	-0.85	-	1.40E-03	1.33E-08	6.31E-03	PDE4D	intron	76.86
15	28205736	(TCTA)n	-2.41	-	1.33E-02	1.52E-08	7.21E-03	OCA2	intron	100
10	12776451	(TA)n|(CA)n	-0.1	-	1.52E-04	1.86E-08	8.82E-03	CAMK1D	intron	87.9
20	4145696	(T)n	-0.33	-	1.80E-04	2.01E-08	9.53E-03	SMOX	intron	25.9
4	98993735	(AAAAT)n	-0.23	-	1.28E-04	2.17E-08	1.03E-02	STPG2	intron	60.93
7	30862243	(TTTA)n	-0.13	-	3.03E-04	2.25E-08	1.07E-02	MINDY4|INMT	intron	99.36
4	10618431	(AT)n	-0.16	-	1.65E-04	2.33E-08	1.11E-02	CLNK	intron	27.6
1	97596334	(TTC)n*|(TCC)n*	13.79	+	9.82E-01	2.38E-08	1.13E-02	DPYD-AS1|DPYD	intron	29.72
20	32178985	(A)n	-0.15	-	8.99E-04	2.56E-08	1.21E-02	CBFA2T2	intron	29.51
6	87683034	(TG)n	1.94	+	9.66E-01	2.61E-08	1.24E-02	HTR1E	intron	97.88
22	18151796	(T)n	-0.5	-	9.97E-04	2.70E-08	1.28E-02	BCL2L13	intron	20.81
7	154128537	(TTTTC)n	0.18	+	2.90E-04	3.09E-08	1.47E-02	DPP6	intron	13.8
17	4776912	(T)n	-0.36	-	1.13E-02	3.14E-08	1.49E-02	MINK1	intron	27.6
9	129708812	(A)n	0.26	+	6.20E-02	3.20E-08	1.52E-02	RALGPS1	intron	22.29
15	52889386	(ATTCT)n	-0.11	-	2.07E-04	3.41E-08	1.62E-02	FAM214A	intron	98.73
9	9980624	(A)n	0.2	+	1.12E-03	3.67E-08	1.74E-02	PTPRD	intron	90.23
2	144392763	(T)n	-15.46	-	9.85E-01	3.93E-08	1.86E-02	ARHGAP15	intron	36.31
1	117852225	(AAAG)n	0.42	+	1.03E-03	3.96E-08	1.88E-02	LINC01525	intron	25.27
5	139247358	(GT)n	0.07	+	3.01E-02	4.37E-08	2.07E-02	NRG2	intron	56.48
8	114124640	(AT)n	0.07	+	2.29E-04	4.69E-08	2.22E-02	CSMD3	intron	14.65
9	113006579	(TA)n|(CA)n	0.06	+	1.78E-04	5.03E-08	2.39E-02	TXN	intron	74.1
5	170024801	(TA)n	0.08	+	1.95E-04	5.15E-08	2.44E-02	KCNIP1	intron	46.28
5	147589560	(CGTA)n*	0.29	+	1.77E-04	5.47E-08	2.59E-02	SPINK6	intron	87.05
12	8197814	(T)n	-0.17	-	1.96E-03	5.65E-08	2.68E-02	FOXJ2	intron	61.57
6	11726132	(A)n	0.13	+	1.08E-04	5.88E-08	2.79E-02	ADTRP	intron	45.44
7	21710485	(GT)n	0.32	+	5.79E-04	6.01E-08	2.85E-02	DNAH11	intron	97.03
4	162704799	(CT)n|(TT)n	-0.22	-	6.36E-04	6.29E-08	2.98E-02	FSTL5	intron	97.66
2	71876267	(A)n	-1.22	-	1.50E-03	6.35E-08	3.01E-02	DYSF	intron	11.89
18	64255842	(AT)n	0.26	+	4.30E-04	6.42E-08	3.05E-02	CDH19	intron	81.74
19	19156302	(T)n	0.73	+	3.28E-01	6.59E-08	3.13E-02	ARMC6	intron	12.1
22	18067188	(GAAA)n|(GAAG)n	0.15	+	1.79E-04	7.34E-08	3.48E-02	LOC101929372|SLC25A18	intron	91.72
23	106465140	(ATA)n	0.33	+	5.53E-05	7.55E-08	3.58E-02	PIH1D3	intron	90.66
22	39009419	(TA)n*	0.01	+	2.75E-02	7.74E-08	3.67E-02	FAM227A	intron	27.39
3	155343999	(A)n	0.05	+	2.08E-04	8.02E-08	3.80E-02	PLCH1	intron	99.79
2	48826033	(A)n	0.41	+	2.31E-04	8.05E-08	3.82E-02	STON1-GTF2A1L	intron	5.52
17	32371125	(CA)n|(CG)n	0.07	+	2.63E-04	8.25E-08	3.91E-02	ASIC2	intron	93.63

12	15660384	(TAAA)n|(TAAC)n	0.22	+	7.67E-05	8.56E-08	4.06E-02	PTPRO	intron	99.79
11	114030659	(G)n	0.17	+	2.96E-02	8.79E-08	4.17E-02	ZBTB16	intron	74.31
18	57361096	(T)n	0.54	+	7.83E-03	8.80E-08	4.17E-02	CCBE1	intron	57.75
1	174403246	(AC)n	-0.11	-	7.71E-02	1.03E-07	4.89E-02	RABGAP1L	intron	57.96

Significant p-values indicate that the distribution of microsatellite lengths in AD group is not identical to that in normal group. However, microsatellite length may affect AD risk in a subtle way. The distributions of microsatellite length in the AKIRIN, TOMM40, MCU, and EML6 genes are shown in Figure 4. Long repeats in the AKIRIN, TOMM40, and MCU genes may increase the risk of AD, whereas in the EML6 the risk increases with short repeats.

3.3 Linkage Disequilibrium between microsatellites and SNPs

Similarly to the T-repeat in the TOMM40 gene, there might be linkage disequilibrium (LD) between microsatellite loci and single nucleotide polymorphism (SNPs) that are in close proximity. If the LD holds, the SNPs around microsatellite loci should be associated with AD. Figure 5 shows that there are many SNPs around the T-repeat in the TOMM40 gene. Those significant SNPs implicitly confirm the strong LD previously found in this region. Less significant SNPs in the AKIRIN2, MCU, and EML6 genes suggest very modest or no LD. Note that two significant SNPs ($p > 10^{-4}$) are on the microsatellite sequence, and their polymorphism may disrupt the very long intact AGAT-repeat.

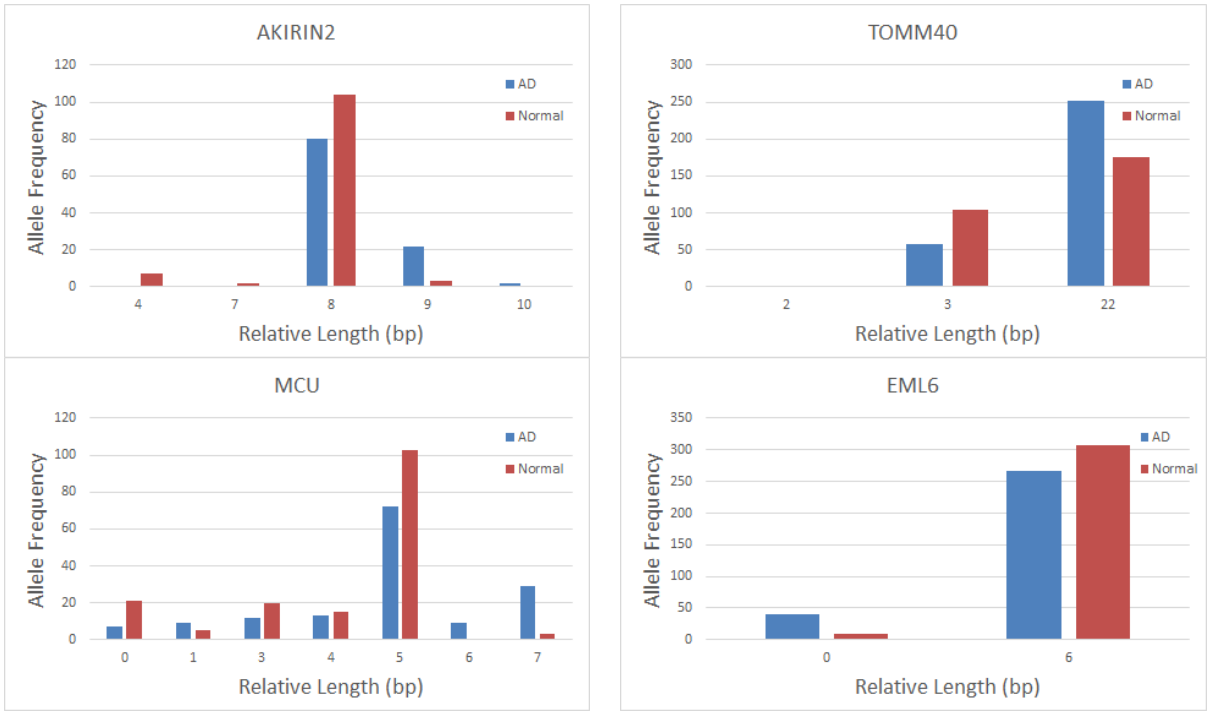


Figure 4. The distributions of microsatellite lengths in the introns of the gene AKIRIN, TOMM40, MCU, and EML6. The repeat units of microsatellites in those genes are (A)n, (T)n, (GT)n, and (AGAT)n, respectively.

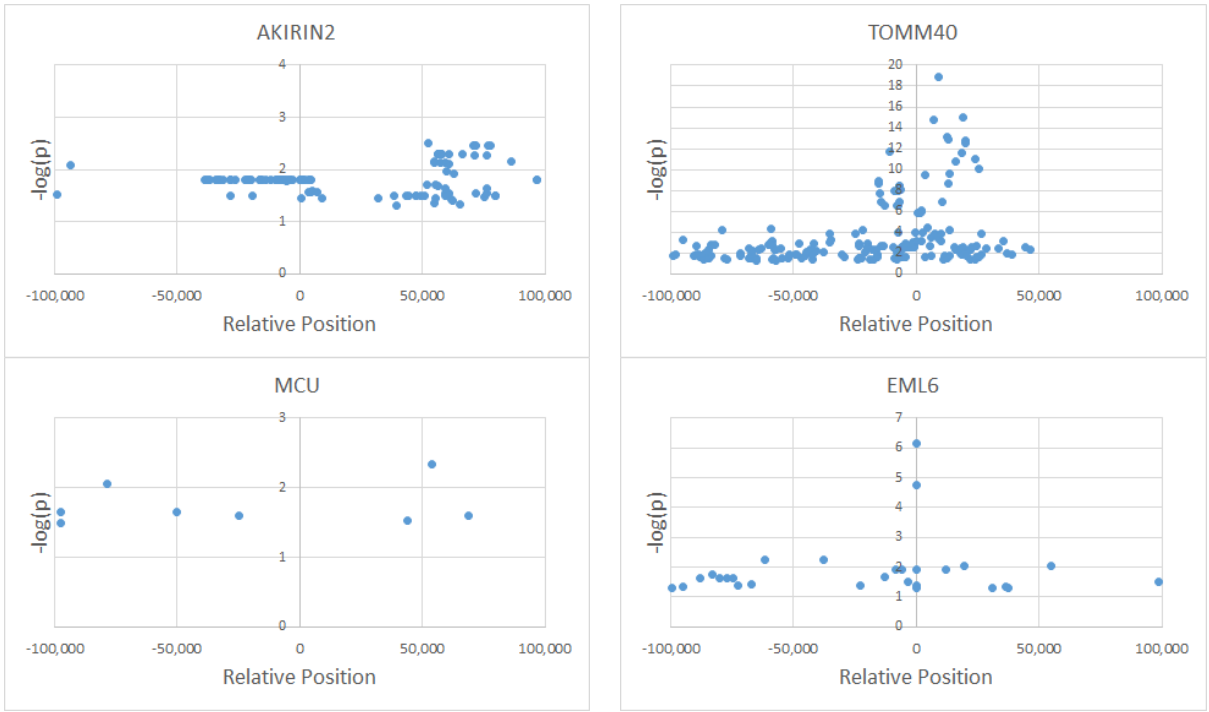


Figure 5. Each dot represents a SNP that is in close proximity with the microsatellite loci in the AKIRIN2, TOMM40, MCU, and EML6 genes. The microsatellite loci are centered at zero on the horizontal axis. The vertical axis is minus logarithm of p-value obtained from the association between a SNP and AD.

3.4 Differential expression of microsatellite-embedded genes in cerebral cortex

We have investigated the differentially expressed (DE) genes in cerebral cortex between AD and normal subjects. The microsatellite loci are not significantly associated with the differential expression of their host genes (OR = 0.72, p = 0.80). The microsatellite-embedded genes that differentially expressed cerebral cortex are shown in Figure 6.

	Significant DE genes	Non-significant DE genes
Significant microsatellite loci	3	67
Non-significant microsatellite loci	959	15,408

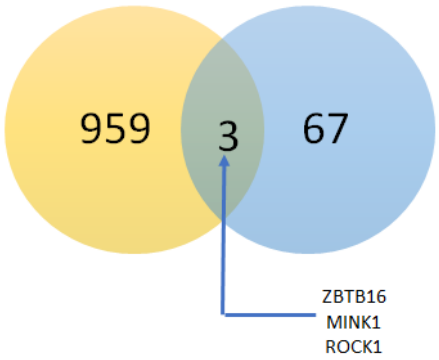


Figure 6. The intersection between the 70 microsatellite-embedded genes and the differentially expressed genes in cerebral cortex.

3.5 Differential expression of microsatellite-embedded genes in temporal cortex

We have investigated the differentially expressed (DE) genes in temporal cortex between AD and normal subjects. Microsatellite-embedded genes are not significantly associated with the differential expression of their host genes (OR = 0.62, p = 0.31). The microsatellite-embedded genes that differentially expressed temporal cortex are shown in Figure 7.

	Significant DE genes	Non-significant DE genes
Significant microsatellite loci	7	63
Non-significant microsatellite loci	2,488	13,879

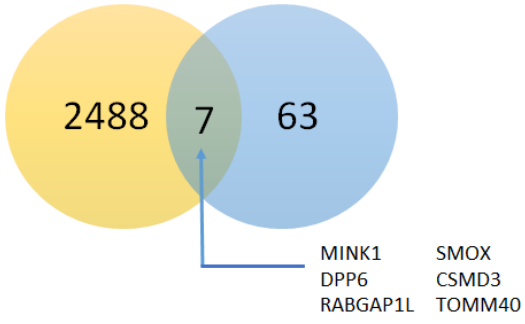


Figure 7. The intersection between the 70 microsatellite-embedded genes and the differentially expressed genes in temporal cortex.

3.6 No isoform switches in Alzheimer's disease

We have searched for isoform switches in both cerebral and temporal cortex by analyzing the Mayo Pilot RNAseq Dataset at transcript level. The microsatellite-embedded transcripts and their expression levels are shown in Figure 8, 9, and 10, and 11. Note that EML6 transcripts have not been found in both cerebral and temporal cortex subjects.

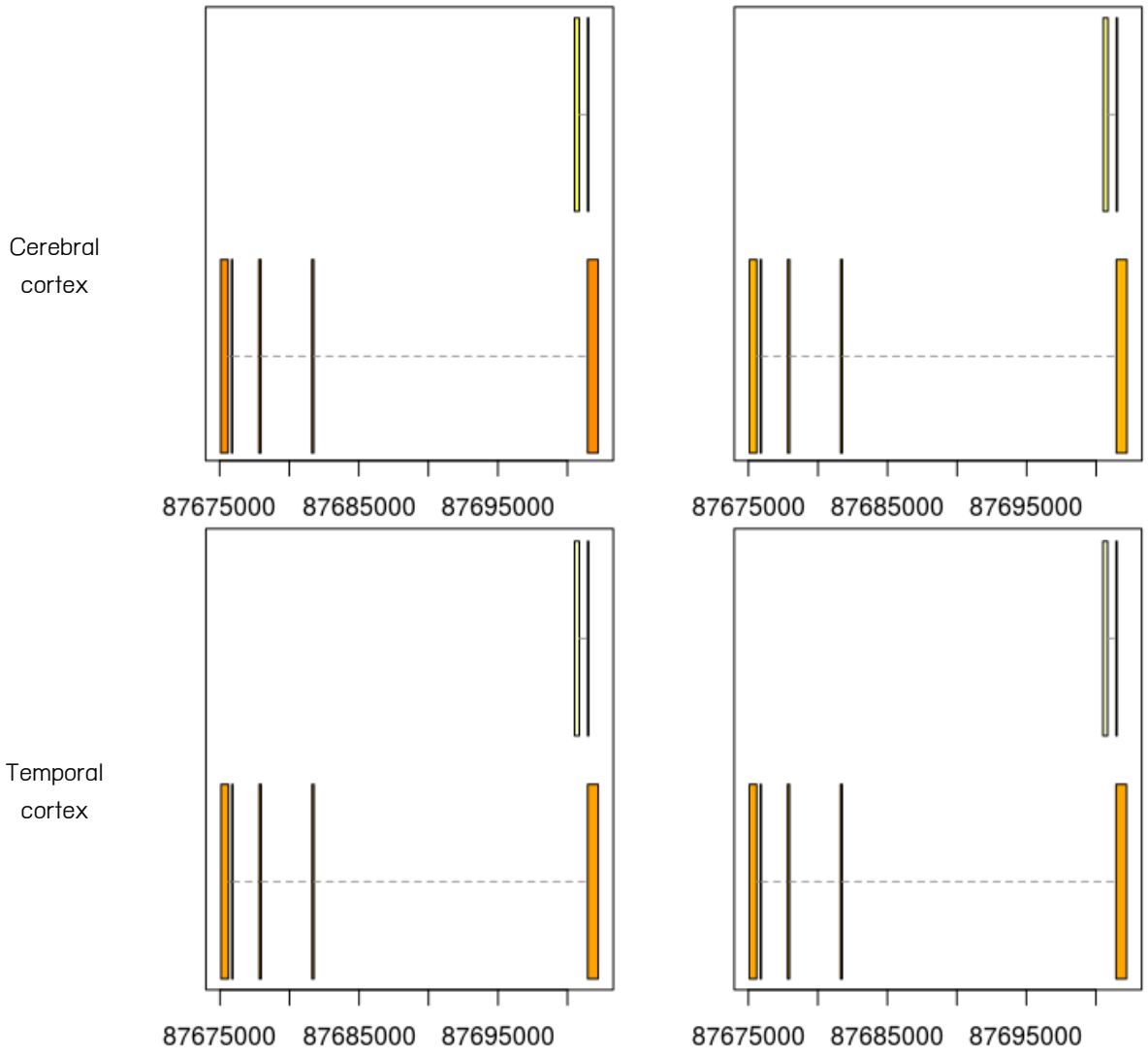


Figure 8. Transcript variants of the AKIRIN2 genes and their expression levels.

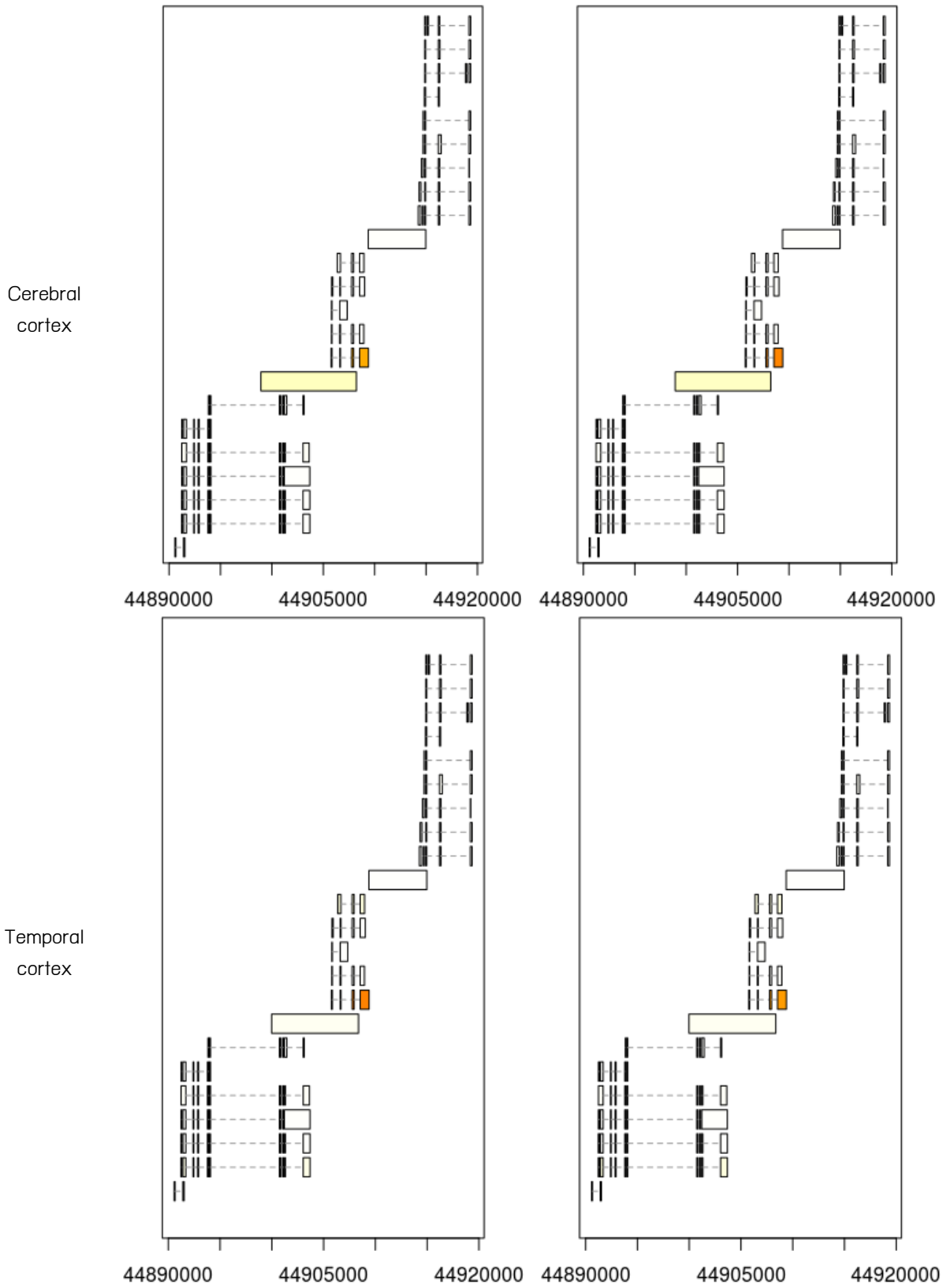


Figure 9. Transcript variants of the TOMM40 genes and their expression levels.

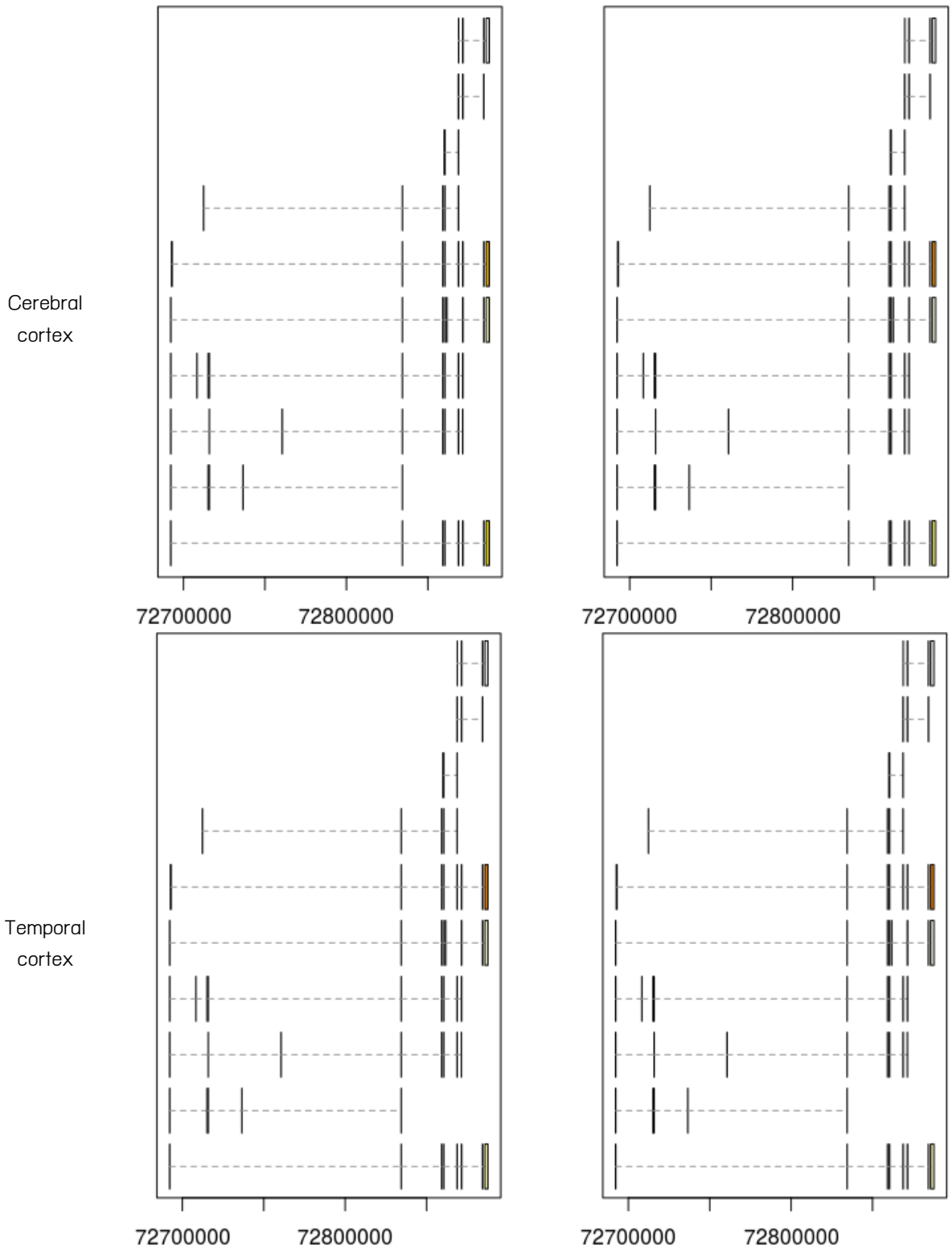


Figure 10. Transcript variants of the MCU genes and their expression levels.

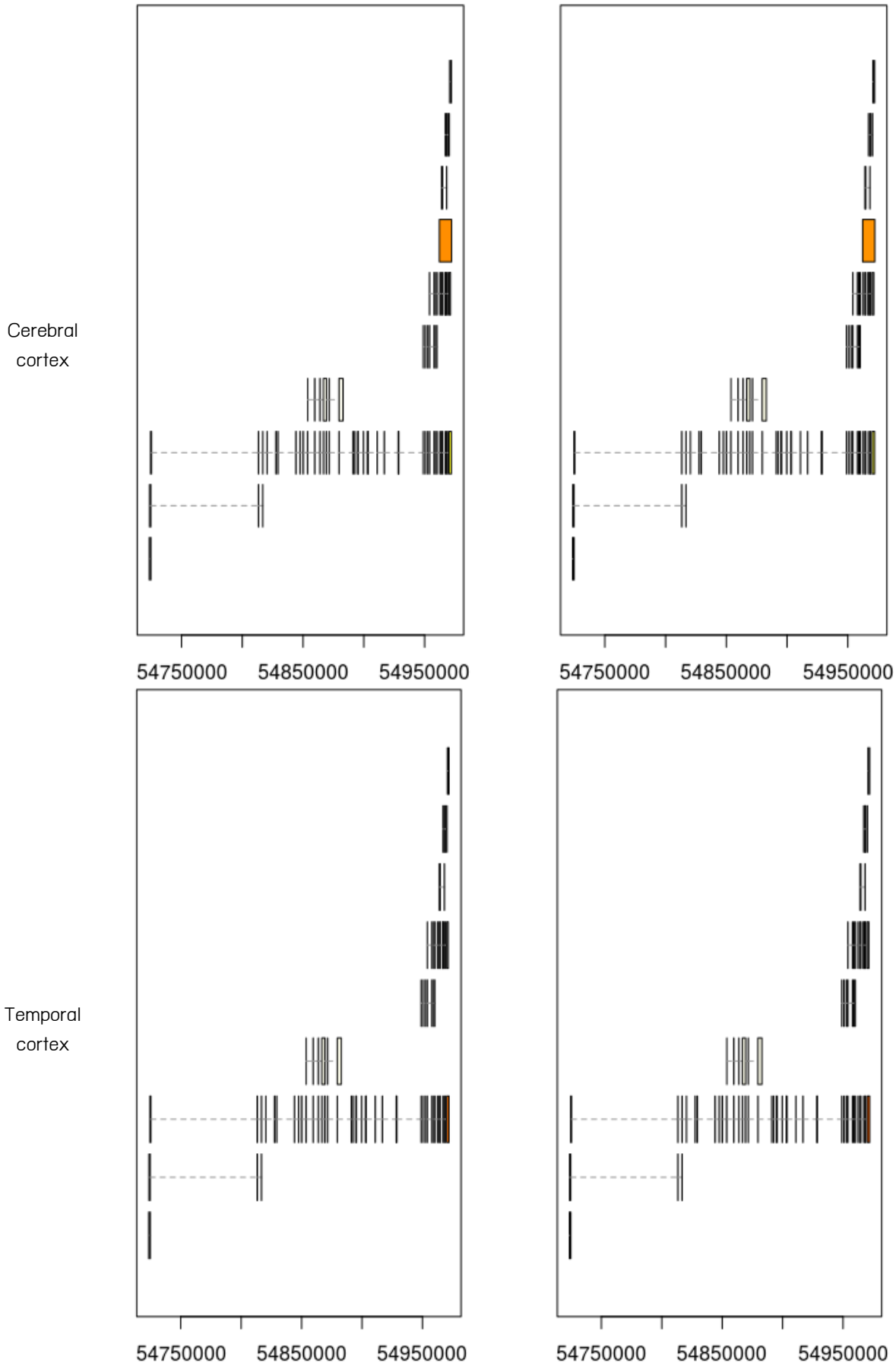


Figure 11. Transcript variants of the EML6 genes and their expression levels.

3.7 Novel transcripts are missing in AD subjects

We detected novel transcripts in cerebral and temporal cortex by means of de-novo transcriptome assembly. These novel transcripts were not annotated in the human transcriptome reference. Moreover, some novel exons and loci were missing in AD subjects, but found in normal subjects. The table below shows the comparison between the de-novo transcriptome assembly and the reference transcriptome.

	Cerebral cortex			
	AD subjects		Normal subjects	
Missed exons	0 / 567213	0.0%	0 / 567213	0.0%
Novel exons	5177 / 577935	0.9%	5941 / 578500	1.0%
Missed introns	2126 / 347401	0.6%	2126 / 347401	0.6%
Novel introns	0 / 347401	0.0%	0 / 347401	0.0%
Missed loci	0 / 57036	0.0%	0 / 57036	0.0%
Novel loci	4719 / 61845	7.6%	5396 / 62678	8.6%

	Temporal cortex			
	AD subjects		Normal subjects	
Missed exons	0 / 567213	0.0%	0 / 567213	0.0%
Novel exons	3946 / 575261	0.7%	5692 / 577504	1.0%
Missed introns	2126 / 347401	0.6%	2126 / 347401	0.6%
Novel introns	0 / 347401	0.0%	0 / 347401	0.0%
Missed loci	0 / 57036	0.0%	0 / 57036	0.0%
Novel loci	3575 / 60863	5.9%	5211 / 62576	8.3%

4. Discussion

The pathogenesis of AD is dominated by the amyloid hypothesis because of the accumulation of amyloid plaques in brains. As a result, previous genetic studies have been focusing on amyloid-related genes such as the APOE gene. APO-ε3 and APO-ε4 alleles, which are composed of two SNPs, are strongly associated with the risk of developing AD. Both SNPs are in exons, and are non-synonymous SNPs. Thus, it is convenient to explain the biological mechanism of amyloid hypothesis. Although the amyloid hypothesis have been firmly established, there are still no proofs that amyloid plaques are the true cause of AD. On the other hand, the discovery of linkage disequilibrium around the APOE gene suggests that the long T-repeat in the TOMM40 gene might be the true risk factor. This because the long T-repeat is a single risk allele at a single locus, whereas APO-ε3 and APO-ε4 are two risk alleles at two loci. Note that the long T-repeat is also associated with APO-ε3 and APO-ε4. The Occam's razor suggests the T-repeat because it is a shorter hypothesis. Although the biological function of TOMM40 protein is relevant to the genesis of AD, it is not easy to elucidate the biological function of T-repeat which is in the intron of TOMM40 gene.

Our genome-wide scan of microsatellites in AD subjects shows that the T-repeat is not the only microsatellite that is associated with the risk of AD. Other classes of microsatellite such as di- and tetra-nucleotide repeats may be implicated in the development of AD. The genome-wide scan found the

microsatellite repeat in the TOMM40 gene as well as in the other genes such as the AKIRIN2, MCU, and EML6 genes. These genes are biologically relevant to AD.

- Akirin2 is essential for the formation of the cerebral cortex. Akirin2 KD in mice results in early embryonic lethal. Analyzing control and knockout transcriptomes using RNA sequencing suggests that Akirin2 is critical for activating genes maintaining progenitor fate, and for repressing the genes associated with neuronal differentiation [28,29].
- MCU encodes mitochondrial inner membrane calcium uniporter that mediates calcium uptake into mitochondria. The imbalance of calcium ions in the mitochondria may contribute to neurodegenerative diseases. A number of studies suggest that the alteration of calcium ions homeostasis is a hallmark of AD [30,31].
- EML6 encodes a protein in a family of microtubule-associated proteins (MAPs). This protein is found on microtubules and regulate microtubule dynamics. EML mutations are found in neuronal disorders and oncogenic fusions in human cancers [32]. We postulated that the EML6 gene may be involved in the collapse of microtubules and forming neurofibrillary tangles observed in AD patients.

Although the functions of microsatellite-embedded genes are pertinent to the AD pathogenesis, it is obscure to conclude that the microsatellites are truly genetic risk factors. There might be two possibilities. Firstly, SNPs in LD with microsatellite loci are the true genetic risk factors. However, highly significant SNPs are only found in the cluster of the APOE, TOMM40, and APOC1 genes. GWAS could not detect modest-effect SNPs individually, but a microsatellite marker that represents its local haplotype would be an easier target. Secondly, the microsatellites are the true genetic risk factors. A mutation in intronic microsatellites can modulate gene expression and alternative splicing. Moreover, expanded repeats produce toxic RNAs that disrupt RNA-binding proteins. However, more studies are needed to confirm that microsatellites are one of the genetic risk factors. As WGS is becoming a standard practice, genome-wide scans of microsatellites in other neurodegenerative disorders will unravel the secret of repetitive non-coding sequences which in the past were presumably junk DNA.

Despite the limited knowledge to reach a conclusion, microsatellite markers are immediately useful for predicting the risk of development and the age of disease onset. We believe that the prediction accuracy can be improved by combining multiple microsatellite markers. In addition, multiple markers are more robust when a population is composed of different ethnic groups.

References

1. Cummings JL. Alzheimer's disease. N. Engl. J. Med. 2004 Jul:351:56-57.
2. Alzheimer's disease - plaques, tangles, causes, symptoms & pathology. youtu.be/v5gdH_Hydes. 2016.
3. World Alzheimer Report 2015. Alzheimer's Disease International (ADI). 2015.
4. Alzheimer's Disease Genetics Fact Sheet. National Institute for Aging. U.S. Department of Health & Human Services. 2015.
5. Bird TD. Genetic aspects of Alzheimer disease. Genet Med. 2008 Apr;10(4):231-9.
6. Guerreiro RJ, Gustafson DR, Hardy J. The genetic architecture of Alzheimer's disease: beyond APP, PSENs and APOE. Neurobiol Aging. 2012 Mar;33(3):437-56.
7. Ellegren H. Microsatellites: simple sequences with complex evolution. Nat Rev Genet. 2004 Jun;5(6):435-45.

8. Brouwer JR, Willemsen R, Oostra BA. Microsatellite repeat instability and neurological disease. Bioessays. 2009 Jan;31(1):71-83.
9. Budworth H, McMurray CT. A brief history of triplet repeat diseases. Methods Mol Biol. 2013;1010:3-17.
10. Lannfelt L, Lilius L, Viitanen M, Houlden H, Rossor M, Hardy J, Winblad B, Basun H. Microsatellite D21S210 (GT-12) allele frequencies in sporadic Alzheimer's disease. Acta Neurol Scand. 1995 Feb;91(2):145-8.
11. Bertram L, Saunders AJ, Mullin K, Sampson A, Moscarillo TJ, Basset SS, Go RC, Blacker D, Tanzi RE. No association between marker D10S1423 and Alzheimer's disease. Mol Psychiatry. 2003 Jun;8(6):571-3.
12. Gohlke H, Illig T, Klopp N, Wagenpfeil S, Konta L, Laws SM, Kurz A, Riemenschneider M. Association study between the D10S1423 microsatellite marker and Alzheimer's disease. Neurobiol Aging. 2006 May;27(5):776.e1-776.e3.
13. Wang LZ, Tian Y, Yu JT, Chen W, Wu ZC, Zhang Q, Zhang W, Tan L. Association between late-onset Alzheimer's disease and microsatellite polymorphisms in intron II of the human toll-like receptor 2 gene. Neurosci Lett. 2011 Feb 11;489(3):164-7.
14. Li Y, Seidel K, Marschall P, Klein M, Hope A, Schacherl J, Schmitz J, Menk M, Schefe JH, Reinemund J, Hugel R, Walden P, Schlosser A, Volkmer R, Schimkus J, Kölsch H, Maier W, Kornhuber J, Frölich L, Klare S, Kirsch S, Schmerbach K, Scheele S, Grittner U, Zollmann F, Goldin-Lang P, Peters O, Kintscher U, Unger T, Funke-Kaiser H. A polymorphic microsatellite repeat within the ECE-1c promoter is involved in transcriptional start site determination, human evolution, and Alzheimer's disease. J Neurosci. 2012 Nov 21;32(47):16807-20.
15. Roses AD, Lutz MW, Amrine-Madsen H, Saunders AM, Crenshaw DG, Sundseth SS, Huentelman MJ, Welsh-Bohmer KA, Reiman EM. A TOMM40 variable-length polymorphism predicts the age of late-onset Alzheimer's disease. Pharmacogenomics J. 2010 Oct;10(5):375-84.
16. Roses AD. An inherited variable poly-T repeat genotype in TOMM40 in Alzheimer disease. Arch Neurol. 2010 May;67(5):536-41.
17. Lin MT, Beal MF. Mitochondrial dysfunction and oxidative stress in neurodegenerative diseases. Nature. 2006 Oct 19;443(7113):787-95.
18. Petersen RC, Aisen PS, Beckett LA, Donohue MC, Gamst AC, Harvey DJ, Jack CR Jr, Jagust WJ, Shaw LM, Toga AW, Trojanowski JQ, Weiner MW. Alzheimer's Disease Neuroimaging Initiative (ADNI): clinical characterization. Neurology. 2010 Jan 19;74(3):201-9.
19. Weiner MW, Veitch DP, Aisen PS, Beckett LA, Cairns NJ, Green RC, Harvey D, Jack CR, Jagust W, Liu E, Morris JC, Petersen RC, Saykin AJ, Schmidt ME, Shaw L, Shen L, Siuciak JA, Soares H, Toga AW, Trojanowski JQ; Alzheimer's Disease Neuroimaging Initiative. The Alzheimer's Disease Neuroimaging Initiative: a review of papers published since its inception. Alzheimers Dement. 2013 Sep;9(5):e111-94.
20. Lim KG, Kwoh CK, Hsu LY, Wirawan A. Review of tandem repeat search tools: a systematic approach to evaluating algorithmic performance. Brief Bioinform. 2013 Jan;14(1):67-81.
21. Balding DJ. A tutorial on statistical methods for population association studies. Nat Rev Genet. 2006 Oct;7(10):781-91.
22. Scholz F, Zhu A. Package 'kSamples'. 2018 Jun.
23. Allen M, Carrasquillo MM, Funk C, Heavner BD, Zou F, Younkin CS, Burgess JD, Chai HS, Crook J, Eddy JA, Li H, Logsdon B, Peters MA, Dang KK, Wang X, Serie D, Wang C, Nguyen T, Lincoln S, Malphrus K, Bisceglio G, Li M, Golde TE, Mangravite LM, Asmann Y, Price ND, Petersen RC, Graff-Radford NR, Dickson DW, Younkin SG, Ertekin-Taner N. Human whole genome genotype and

transcriptome data for Alzheimer's and other neurodegenerative diseases. Sci Data. 2016 Oct 11;3:160089.
24. Pertea M, Kim D, Pertea GM, Leek JT, Salzberg SL. Transcript-level expression analysis of RNA-seq experiments with HISAT, StringTie and Ballgown. Nat Protoc. 2016 Sep;11(9):1650-67.
25. Magis AT, Funk CC, Price ND. SNAPR: a bioinformatics pipeline for efficient and accurate RNA-seq alignment and analysis. IEEE Life Sci Lett. 2015 Aug;1(2):22-25.
26. Huang da W, Sherman BT, Lempicki RA. Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources. Nat Protoc. 2009;4(1):44-57.
27. Huang da W, Sherman BT, Lempicki RA. Bioinformatics enrichment tools: paths toward the comprehensive functional analysis of large gene lists. Nucleic Acids Res. 2009 Jan;37(1):1-13.
28. Bosch PJ, Fuller LC, Sleeth CM, Weiner JA. Akirin2 is essential for the formation of the cerebral cortex. Neural Dev. 2016 Nov 21;11(1):21.
29. JA Weiner. Molecular mechanisms of neuronal differentiation and neural circuit formation. biology.uiowa.edu/people/joshua-weiner. 2018.
30. Calì T, Ottolini D, Brini M. Mitochondrial Ca(2+) and neurodegeneration. Cell Calcium. 2012 Jul;52(1):73-85.
31. Liao Y, Dong Y, Cheng J. The Function of the Mitochondrial Calcium Uniporter in Neurodegenerative Disorders. Int J Mol Sci. 2017 Feb 10;18(2). pii: E248.
32. Fry AM, O'Regan L, Montgomery J, Adib R, Bayliss R. EML proteins in microtubule regulation and human disease. Biochem Soc Trans. 2016 Oct 15;44(5):1281-1288.

Research Outcomes

This research project has been published in the following journals.

Journals (no status indicates that the paper has been already published)

1. Aporntewan C, Pin-on P, Chaiyaratana N, Pongpanich M, Boonyaratanakornkit V, Mutirangura A. Upstream mononucleotide A-repeats play a cis-regulatory role in mammals through the DICER1 and Ago proteins. Nucleic Acids Res. 2013 Oct;41(19):8872-85. doi: 10.1093/nar/gkt685. Epub 2013 Aug 8. PubMed PMID: 23935075; PubMed Central PMCID: PMC3799445.
2. Aporntewan C, Mutirangura A. Gene Ontology-Based Analysis Reveals a Physiological Role of Upstream Mononucleotide A-repeats in Mammals (complete manuscript to be published)

Appendix

A collection of publications due to RSA5980060 grant is listed below.

Journals

1. Aporntewan C., Chongstitvatana P., Chaiyaratana N. Indexing Simple Graphs by Means of the Resistance Distance, IEEE Access, Vol. 4, pp. 5570-78, September 2016.
2. Aporntewan C., Jaikaew P., Mutirangura A. Genome-wide scan of microsatellites associated with Alzheimer's disease (complete manuscript to be published)

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 7, 2016, accepted August 24, 2016, date of publication September 7, 2016, date of current version October 6, 2016.

Digital Object Identifier 10.1109/ACCESS.2016.2606764

Indexing Simple Graphs by Means of the Resistance Distance

CHATCHAWIT APORNTEWAN[1], PRABHAS CHONGSTITVATANA[2], (Member, IEEE), AND NACHOL CHAIYARATANA[3], (Member, IEEE)

[1]Department of Mathematics and Computer Science & Omics Science and Bioinformatics Center, Faculty of Science, Chulalongkorn University, Bangkok 10330, Thailand
[2]Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok 10330, Thailand
[3]Department of Electrical and Computer Engineering, Faculty of Engineering, King Mongkut's University of Technology North Bangkok, Bangkok 10800, Thailand

Corresponding author: C. Aporntewan (chatchawit.a@chula.ac.th)

This work was supported in part by the Thailand Research Fund under Grant RSA5980060 and in part by the Chulalongkorn Academic Advancement into Its 2nd Century Project (CUAASC).

ABSTRACT For every simple connected graph, we present a polynomial time algorithm for computing a numerical index, which is composed of primary and secondary parts. Given a graph $G = (V, E)$ where V and E are, respectively, vertex and edge sets, the primary part of the index is a set of $|V|$ fractions and the secondary part of the index is a set of $|B| \times |V|$ fractions, where B is the partition of the vertex set V. Basically, each fraction in the primary and secondary parts is the electrical resistance between two vertices when every edge in the graph is replaced with a unit resistor (1 Ω). The experimental results show that our indexing algorithm produced a unique index for every simple connected graph with $\leq$10 vertices, including all graphs that are counterexamples for detecting graph isomorphism by resistance spectrum comparison. The strength of our indexing algorithm lies in its extreme simplicity. An index of a graph is solely derived from the determinants of reduced Laplacian matrices, which represent the graph. Therefore, the performance of our indexing algorithm only depends on how fast the matrix determinants can be computed.

INDEX TERMS Electrical resistance, graph indexing, graph isomorphism, resistance distance, simple connected graphs.

I. INTRODUCTION

Graph is a data structure that has been used for representing data in a wide range of applications including an EXtensible Markup Language (XML), chemical compounds, multimedia databases, social networks, biological pathways, protein-protein interaction networks, semantic webs, and business process models [1]. The increasing popularity in these applications produces a plethora of graph databases which demand an efficient querying method. A graph database could be either a single extremely large graph (e.g., social networks) or a large collection of small graphs (e.g., chemical compounds). Basically there are three types of query in graph databases.

- *Exact matching query.* The task is to search for graphs in a database which are exactly matched or isomorphic with the query graph. Let V and E respectively represent vertex and edge sets in a graph. Graphs $G = (V_G, E_G)$ and $H = (V_H, E_H)$ are isomorphic if and only if there exists a permutation matrix P such that $A_H = PA_GP^{-1}$ where A_G and A_H are respectively the adjacency matrices that represent graphs G and H [2].
- *Subgraph/supergraph query*. This task is to search for graphs in a database in which the query graph is a subgraph or a supergraph. It relates to the subgraph isomorphism problem which determines whether graph G contains a subgraph that is isomorphic to graph H. This problem is known to be NP-complete [3].
- *Similarity (approximate matching) query*. This task is to search for graphs in a database which share some similarities with the query graph. The degree of similarity is defined by edit distance between two graphs [4].

Although the exact matching query, which searches for isomorphic graphs, is the simplest one, a polynomial time algorithm for the task has not been found. The fastest known algorithm, standing for more than three decades, has the time complexity $2^{O(\sqrt{n \log n})}$ where n is the number of vertices [5]. Thus, it is still interesting to limit our scope to the exact matching query.

We propose a graph index which is a numerical array of fractions. The indexing algorithm is carried out by replacing every edge in a graph with a unit resistor (1 Ω). The graph can then be viewed as an electronic circuit with some electrical properties. Finally, the graph index is constructed from the resistance characteristics of the circuit. Our indexing algorithm is empirically proven to be perfect (no collisions between non-isomorphic graphs) for all simple connected graphs with $\leq$10 vertices. The algorithm is not limited to only connected graphs because a disconnected graph can be decomposed into multiple connected subgraphs. The index of each subgraph can be computed independently and can be merged later into a single index.

The computational time for computing an index grows in a polynomial relationship with the number of vertices. For larger graphs with $>$10 vertices, although there might be some collisions, we did not find a counterexample – two non-isomorphic graphs that produce the same index. Since finding such a counterexample is not trivial, we estimate that our algorithm is effective for indexing graphs in general.

Nauty and its variant Traces are outstanding algorithms for canonical labeling of graphs [6]. These algorithms relabel vertices in such a way that isomorphic graphs become identical after canonical labeling. Determining the isomorphism of canonized graphs is a direct comparison in a convenient representation, for instance, comparing their adjacency matrices. Moreover, a canonized graph is guaranteed to have a unique index for exact matching queries. Nauty uses a search-tree approach for canonical labeling. At the root of the search tree, the first vertex is chosen for labeling. Branches from the root are choices of the second vertex for labeling and so on. The relabeled graphs appear at leaf nodes of the search tree and only one of them is canonical labeling. Nauty dramatically speeds up the search by exploiting a graph automorphism – the isomorphism of a graph to itself. An automorphism of a graph G is a permutation P such that $A_G = PA_GP^{-1}$ where A_G is the adjacency matrix of graph G. With the innovative use of automorphisms, Nauty avoids an exhaustive search by pruning the search tree. Traces is a variance of Nauty with a major improvement in performance. Although the time complexity of Traces is not bounded by a polynomial, in practice Traces outperforms other algorithms [6]. The performance of Traces is not consistently uniform; it varies significantly with different graph families.

Alternatively, an undirected graph can be indexed using eigenvalues of its adjacency matrix. Unfortunately, two non-isomorphic graphs may produce the same set of eigenvalues. The smallest counterexample is a pair of non-isomorphic connected graphs with six vertices [7]. However, there are polynomial time algorithms for a special case of isomorphism testing where the eigenvalues of an undirected graph have bounded multiplicity. Deterministic and Las Vegas algorithms respectively have time complexity $O(n^{4m+c})$ and $O(n^{2m+c})$ where n is the number of vertices, m is the multiplicity of eigenvalues, and c is an absolute constant [8]. In practice, bounded multiplicity might be too restricted for general use.

Other graph indexing algorithms do not focus only on an exact matching query but are more concerned with subgraph/supergraph and similarity queries. This kind of indexes is made of local structures such as paths in GraphGrep [9], trees in GCoding [10], and subgraphs in GDIndex [11]. The local structures of graphs are more difficult to manipulate than numerical indexes and the size of local-structure indexes may increase drastically with the size of database. Some indexing methods are designed for specific applications, for instance, GString considers the semantics of chemical structures and uses them as index features [12]. A common querying strategy is to use indexes for filtering candidate graphs which are related to the query. Next, each candidate is verified that it really satisfies the conditions of the query. This line of research was reviewed elsewhere [1].

The resistance distance was proposed by Klein and Randić. ''If fixed resistors are assigned to each edge of a connected graph, then the effective resistance between pairs of vertices is a graphical distance'' [13]. This novel distance function has established a number of graph theorems and successful applications in cyclicity, which is a structural feature in graphs [14]–[16]. An efficient method for calculating effective resistance between any two vertices is needed to compute the resistance distance. A common method in electrical engineering is Nodal Anaylsis [17]. However, there is a more efficient way to calculate the resistance between all vertex pairs at once. Given a graph $G = (V, E)$, the following matrix operations result in a resistance-distance matrix Ω, whose element Ω_{ij} is the effective resistance between vertex $v_i \in V$ and vertex $v_j \in V$. Let $L = (l_{ij})$ be the Laplacian matrix of graph G. It is similar to the adjacency matrix except that the diagonal element l_{ii} is equal to the degree of vertex v_i, elements $l_{ij} = l_{ji}$, and element $l_{ij} = -1$ if there is an edge between vertices v_i and v_j. Let Φ be an auxiliary matrix whose all elements are equal to one. Consequently,

$$\Gamma' = [L + 1/|V| \times \Phi]^{-1} \tag{1}$$

and

$$\Omega_{ij} = \Gamma'_{ii} - 2\Gamma'_{ij} + \Gamma'_{jj} \tag{2}$$

where $|V|$ is the number of vertices in graph G. A more elegant method for calculating the effective resistance is solely derived from determinants of reduced Laplacian matrices [18]. The effective resistance is given by

$$\Omega_{ij} = \frac{\det L(i,j)}{\det L(i)} \tag{3}$$

where $L(i)$ is the matrix resulting from removing the i^{th} row and the i^{th} column of Laplacian matrix L, $L(i,j)$ is the matrix resulting from removing both the i^{th} and j^{th} rows as well as the i^{th} and j^{th} columns of Laplacian matrix L, and det denotes a determinant operation.

An example of the calculation of effective resistance using Equation 3 is given in Fig. 1. It is noted that $\det L(i)$ equals

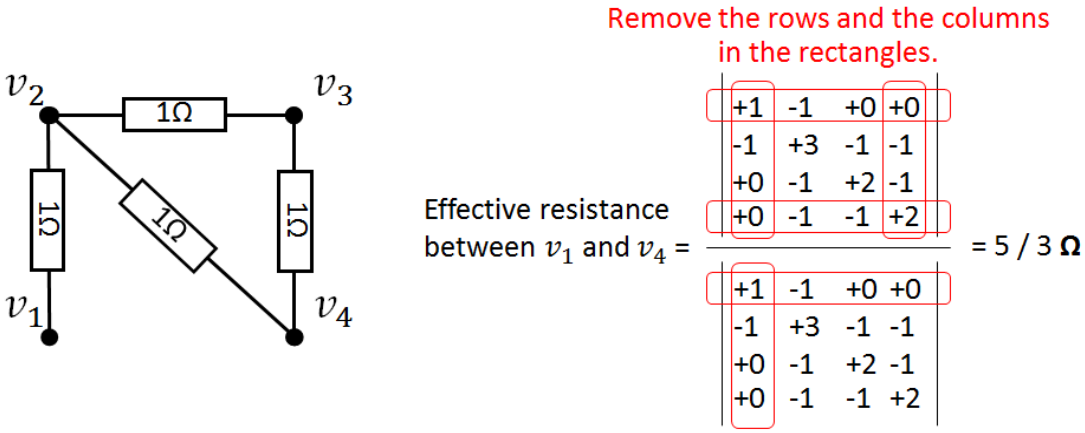


FIGURE 1. An example shows the calculation of effective distance between vertices v_1 and v_4.

the number of spanning trees in the graph represented by Laplacian matrix L. Equations 1, 2, and 3 are suitable for connected graphs. In the case of disconnected graphs, the matrix inversion in Equation 1 cannot be performed while $\det L(i) = 0$ leads to a division by zero in Equation 3. To cope with disconnected graphs, effective resistance Ω_{ij} must be set to $+\infty$ when vertices v_i and v_j are not connected.

A resistance spectrum refers to a set of effective resistance between every pair of vertices in a graph. The original idea of solving the graph isomorphism problem by means of resistance spectrums was discussed elsewhere [19]. A simple algorithm replaces every edge in a graph with a 1-Ω resistor and calculates the resistance spectrum. It was hypothesized that two graphs are isomorphic if and only if their resistance spectrums are identical. However, this hypothesis was rejected quickly after the discovery of counterexamples [20], [21]. Fig. 2 shows 13 pairs of non-isomorphic graphs that every pair produces the same resistance spectrum given in Table 1.

Our indexing algorithm uses an approach similar to that leading to resistance spectrums. However, it is further improved by multiple steps of graph perturbation and resistance measurement. In the Results, it will be shown that our algorithm produced a unique index for every counterexample in Fig. 2.

II. METHODS

A. INDEXING ALGORITHM

Algorithm 1 computes an index (P, S) of a simple connected graph $G = (V, E)$ in two main steps. The first step computes the primary part P which is a set of $|V|$ fractions. The second step computes the secondary part S which is a set of $|B| \times |V|$ fractions where B is a partition of the vertex set V. An example of the algorithm is given in Fig. 3. There are three major steps.

The first step is to build the primary part of the index. A crucial step is to add a dummy vertex v_0 to the graph, and add $|V|$ edges to connect the dummy vertex v_0 with all original vertices $v_1, \ldots, v_{|V|}$. Next, the Laplacian matrix L that corresponds to the new graph is constructed so that the effective resistance between vertices v_0 and v_i can be calculated using the formula $P_i = \det L(0, i)/\det L(0)$. The fraction P_i is then labeled to the original vertex v_i. Finally, the primary part is composed of $|V|$ fractions in the vertex-ordered list P. The vertex-ordered list P is not yet sorted but it will be sorted at the end of the algorithm. The sorting is to canonize the index so that isomorphic graphs produce the same index.

The second step is to partition the vertex set V by vertex labels in the vertex-ordered list P. Basically all vertices with the same label are put in the same block. The resulting partition is a set of $|B|$ blocks where each block is a set of vertices. In addition, the blocks are sorted so that the vertex label (effective resistance) associated with block b_i is always less than the vertex label associated with block b_j when $i < j$. Sorting the blocks is a preparation for canonizing the secondary part of the index.

The third step is to build the secondary part of the index. It is noted that if there is only one block in the partition B, then

TABLE 1. Resistance spectrums of the graphs.

Graph pair	Resistance spectrum
1	6(1) 6(2) 24(+∞)
2	3(2/3) 6(1) 12(5/3) 6(2) 9(8/3)
3	8(1) 13(2) 12(3) 3(4)
4	8(3/4) 6(1) 4(3/2) 8(7/4) 4(2) 4(11/4) 2(3)
5	6(1) 4(2) 2(3) 24(+∞)
6	8(1) 10(2) 10(3) 6(4) 2(5)
7	4(3/4) 7(1) 4(7/4) 6(2) 4(11/4) 5(3) 2(15/4) 3(4) 1(5)
8	1(1/2) 4(5/8) 6(1) 2(3/2) 8(13/8) 4(2) 1(5/2) 6(21/8) 2(3) 2(29/8)
9	10(2/3) 5(1) 4(4/3) 10(5/3) 2(2) 2(7/3) 3(8/3)
10	4(3/4) 7(1) 4(7/4) 6(2) 4(11/4) 5(3) 2(15/4) 3(4) 1(5)
11	2(1/2) 8(5/8) 4(1) 4(5/4) 8(13/8) 4(2) 4(21/8) 2(3)
12	4(3/4) 18(1) 16(7/4) 22(2) 32(11/4) 24(3) 32(15/4) 18(4) 16(19/4) 8(5)
13	29(1) 38(2) 50(3) 64(4) 78(5) 82(6) 64(7) 26(8) 4(9)

The effective resistance between vertices v_i and v_j are shown in ascending order. $n(a/b)$ denotes n occurrences of the fraction a/b.

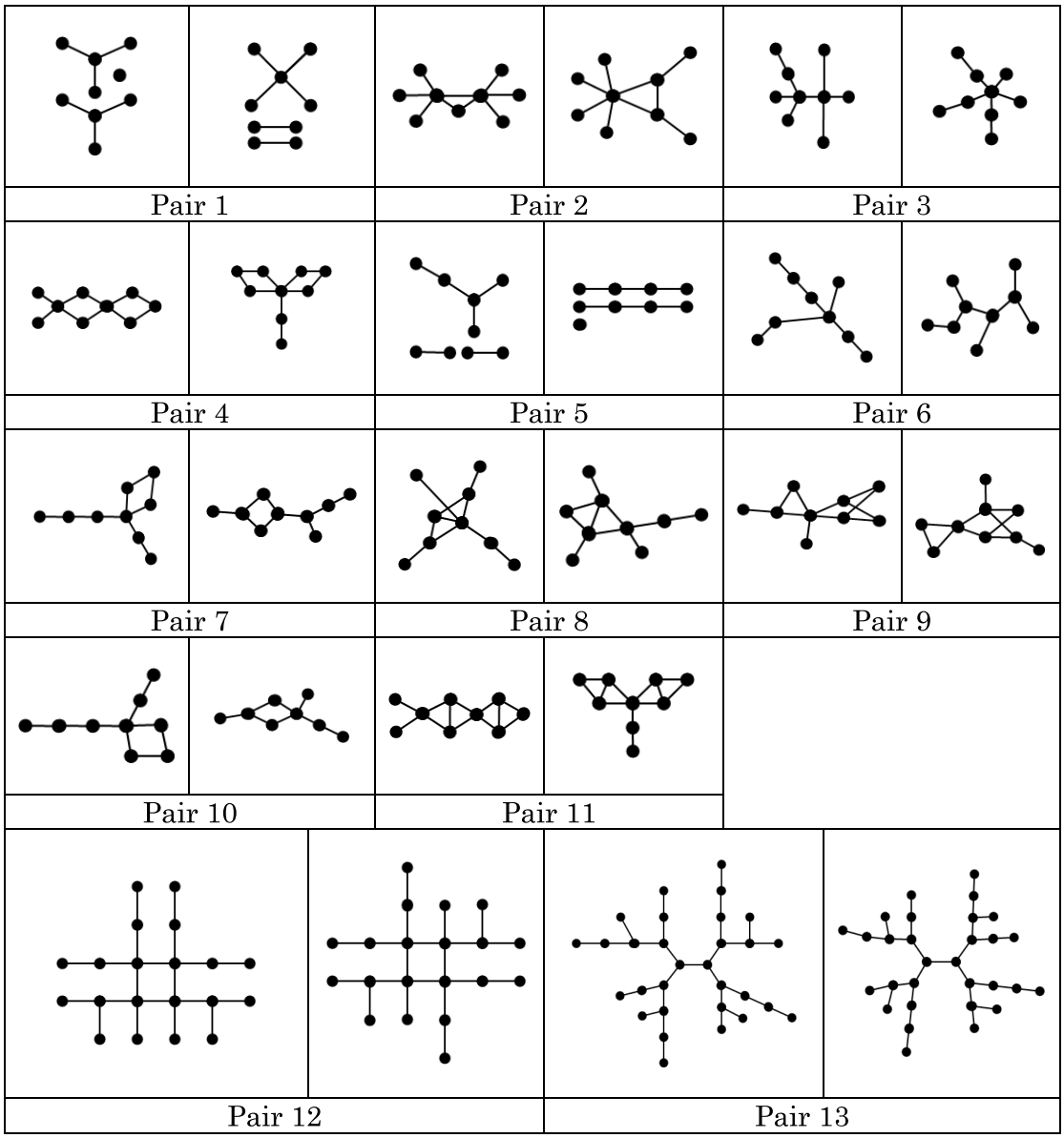


FIGURE 2. Pairs of non-isomorphic graphs that produce the same resistance spectrum.

the secondary part is an empty list. The graph is perturbed using information in the partition B to make the secondary part differs from the primary part. The perturbation is done sequentially for each block b_i where it starts with block b_1 and ends with block $b_{|B|}$ in the following manner. First, a dummy vertex is added and is connected to all vertices in graph G except the vertices in the block b_i. Next, the effective resistance from the dummy vertex v_0 to every original vertex is calculated in a similar fashion to the calculation performed for building the primary part. The effective resistance can reach the value of $+\infty$ if the graph is disconnected. For each perturbation, the effective resistance calculation begins from the original vertices in the first block b_1 and ends at the original vertices in the last block $b_{|B|}$. The resistance values obtained for each block are sorted in ascending order and are appended to the list S. The sorting here is to canonize the secondary part.

Two indexes can be immediately compared with no extra computation. Indexes of graphs G and H are respectively denoted by (P_G, S_G) and (P_H, S_H) where P_G and P_H are the primary parts of the index while S_G and S_H are the secondary parts of the index. Two indexes are exactly matched if and only if the primary parts $P_G[i] = P_H[i]$ for all i and the secondary parts $S_G[j] = S_H[j]$ for all j. In practice, the primary parts are compared first because most non-isomorphic graphs can be distinguished using only the primary parts. In the Results, it will be shown that there were only 1.63% of simple connected graphs with $\leq$10 vertices that

Algorithm 1 Computation of the Primary and the Secondary Parts of the Index

Input: A graph $G = (V, E)$ with $V = \{v_1, \ldots, v_{|V|}\}$.
Output: The primary part P which is an ordered list of $|V|$ fractions and the secondary part S which is an ordered list of $|B| \times |V|$ fractions where B is the partition of the vertex set V.

// building the primary part
P = an empty list of fractions; // P_i denotes the i^{th} element in the list P, $1 \leq i \leq |V|$.
G' = the graph G augmented by a dummy vertex v_0 and edges connecting the dummy vertex v_0 to all vertices in the vertex set V;
L = the Laplacian matrix of graph G' whose rows and columns 0 to $|V|$ correspond to vertices v_0 to $v_{|V|}$;
denominator $= \det L(0)$; // The matrix $L(0)$ is the Laplacian matrix L in which row and column 0 are removed.
for $i = 1$ to $|V|$ do
    numerator $= \det L(0, i)$; // The matrix $L(0, i)$ is the matrix $L(0)$ in which row and column i are removed.
    P_i = the fraction in the lowest terms of $\frac{numerator}{denominator}$;
end
// partitioning the vertex set V
$B = \{b_1, \ldots, b_{|B|}\}$ is a partition of the vertex set V and consists of $|B|$ blocks;
    //Vertices v_i and v_j are in the same block if vertex labels $P_i = P_j$.
    //Vertex labels $P_x < P_y$ if and only if vertices $v_x \in b_i$ and $v_y \in b_j$, and $i < j$.
// building the secondary part
S = an empty list of fractions; // S_i denotes the i^{th} element in the list S, $1 \leq i \leq |B| \times |V|$.
if $|B| = 1$ then
    Sort P in ascending order;
    return (P, S);
end
for $i = 1$ to $|B|$ do // Perturbing graph G in $|B|$ different ways.
    G' = the graph G augmented by a dummy vertex v_0 and edges connecting the dummy vertex v_0 to all vertices in the vertex set $V \setminus b_i$;
    L = the Laplacian matrix of graph G' whose rows and columns 0 to $|V|$ correspond to vertices v_0 to $v_{|V|}$;
    denominator $= \det L(0)$;
    for $b_j \in B$ do
        X = an empty list of fractions.
        for $v_k \in b_j$ do
            numerator $= \det L(0, k)$;
            Append the fraction in the lowest terms of $\frac{numerator}{denominator}$ to the list X;
        end
        Sort X in ascending order;
        Append X to S;
    end
end
Sort P in ascending order;
return (P, S);

their non-isomorphisms must be decided by the secondary parts.

The computation of primary and secondary parts of an index depends mostly on the calculation of matrix determinants. The computation of the primary part calculates $n + 1$ determinants (n numerators, $\det L(0, i)$, and one denominator, $\det L(0)$) where n is the number of vertices excluding the dummy vertex. In the worst case, the partition produces the maximum n blocks and the computation of secondary part calculates $(n + 1)$ determinants for each block. Totally, $(n + 1) + n(n + 1) = (n + 1)^2$ or $O(n^2)$ determinants are calculated for an index of a graph. The determinant of an $n \times n$ matrix can be computed in $O(n^3)$ time using LU decomposition [22]. Therefore, the time complexity of computing an index is $O(n^5)$.

However, the effective resistance is stored as a precise fraction. Therefore, the time complexity also depends on the memory space used for storing fractions. A fraction is composed of a numerator and a denominator, and both are matrix determinants. The largest matrix is $L(0)$, which is the Laplacian matrix reduced by removing the first row and the first column. The determinant of matrix $L(0)$, which is denoted by $\det L(0)$, equals the number of spanning trees in the Laplacian matrix L. Fortunately, many upper bounds for

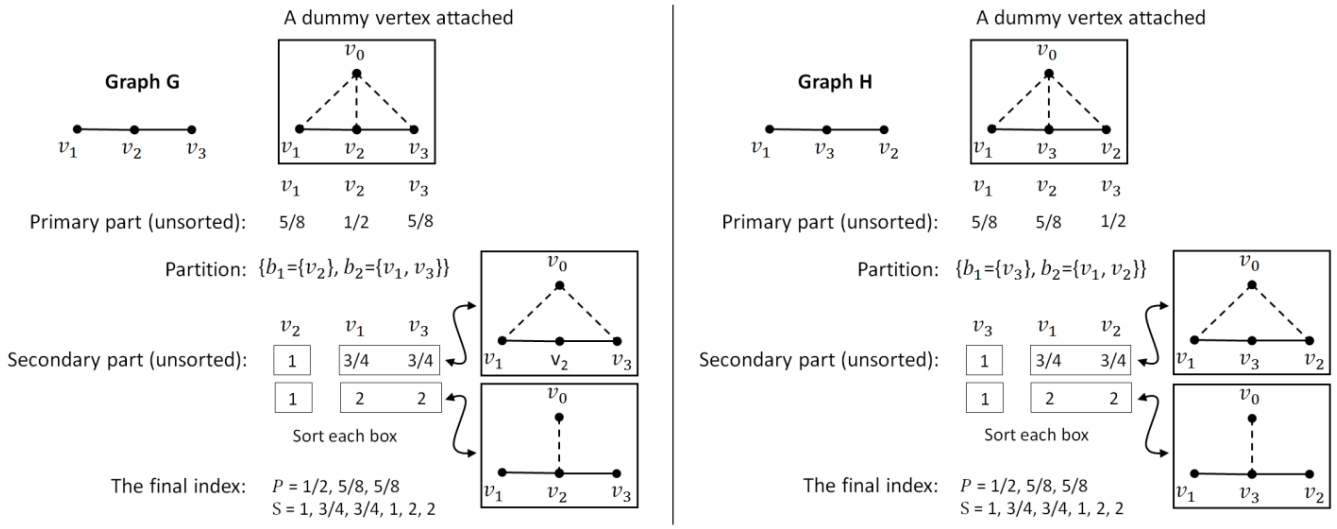


FIGURE 3. An illustration of our indexing algorithm. Two isomorphic graphs *G* and *H* produce the same index.

the number of spanning trees in a graph were proposed [23]. An upper bound for the number of spanning trees in a graph G, denoted by $\kappa(G)$, was proposed by Grimmett [24] and is given by

$$\kappa(G) \leq \frac{1}{n}\left(\frac{2e}{n-1}\right)^{n-1} \tag{4}$$

where n and e are the number of vertices and the number of edges in graph G, respectively. In the worst case, a fully-connected graph has $(n^2-n)/2$ edges. Subsequently, $\kappa(G)$ can be rewritten only in terms of n as

$$\kappa(G) \leq \frac{1}{n}\left(\frac{2\left(\frac{n^2-n}{2}\right)}{n-1}\right)^{n-1} \tag{5}$$

or

$$\kappa(G) \leq n^{n-2} \leq O(n^n). \tag{6}$$

Therefore, $\det L(0)$ is no more than $O(n^n)$. In computer hardware, this number occupies only $O(\log n^n)$ or $O(n \log n)$ bits under the assumption that the implementation of Big Integer in modern programming languages is space efficient. Moreover, the time complexity of basic arithmetic operations such as addition and multiplication increases not faster than a polynomial of the problem size. It is noted that there are sharper bounds for $\kappa(G)$ [23] but using Grimmett's upper bound is sufficient to show that the time complexity of building an index is bounded by a polynomial.

Two isomorphic graphs always produce the same index. However, two graphs with the same index are not necessarily isomorphic. We cannot prove that the index is unique for every simple connected graph. If it can be proved, the graph isomorphism problem will immediately be in the complexity class P. In the Results, it will be shown that the index may be unique for every simple connected graph and the proof of the conjecture is hence worth the pursuit.

B. GRAPH DATASETS

Two datasets were used for benchmarking our indexing algorithm.

- The first dataset was proposed as counterexamples against the use of resistance spectrums for solving the graph isomorphism problem [25]. In this dataset, each pair of non-isomorphic graphs produces the same resistance spectrum. Therefore, the non-isomorphism between two graphs cannot be detected by comparing their resistance spectrums.
- The second dataset contains all simple connected graphs with $\leq$10 vertices. This dataset was taken from McKay's collection of combinatorial data at the Australian National University's website: http://cs.anu.edu.au/~bdm/ data/graphs.html.

C. BIG FRACTION

The effective resistance between vertices v_i and v_j can be written as a fraction of $\det L(i,j)/\det L(i)$. This fraction cannot be stored in a 32-bit or 64-bit floating-point register because the precision will be lost. Thus, we developed a new data structure called Big Fraction. The numerator and the denominator of a Big Fraction are stored separately as Big Integer, which is a data structure for integers with no limitation on the maximum value. The only limitation is the available computer memory. Big Integer is a common data type in modern programming languages such as Java and C#. Every time a new fraction is calculated, the numerator and the denominator are divided by their greatest common divisor (GCD) so that the fraction is always reduced to its lowest terms. Euclidean algorithm makes the calculation of GCD very efficient. LU decomposition

provides a means for the calculation of a Laplacian matrix determinant [22].

III. RESULTS

Our indexing algorithm was tested for correctness using the first two graph datasets described in the Methods. First, we built an index of every counterexample against the use of resistance spectrums for solving the graph isomorphism problem. Second, an index of every simple connected graph with $\leq$10 vertices was generated. Each index was checked whether it collides with the indexes of other non-isomorphic graphs.

TABLE 2. Primary part of the indexes of the graphs.

Graph pair		Index
1	Left	2(2/5) 6(3/5) 1(1)
	Right	1(1/3) 4(7/12) 4(2/3)
2	Left	2(50/187) 1(7/17) 6(106/187)
	Right	1(5/21) 2(22/63) 4(47/84) 2(37/63)
3	Left	1(35/116) 1(9/29) 1(13/29) 2(267/464) 3(67/116) 1(71/116)
	Right	1(5/19) 3(42/95) 2(43/76) 3(58/95)
4	Left	1(35/117) 1(73/234) 2(97/234) 2(17/39) 1(53/117) 2(541/936)
	Right	1(35/136) 4(1223/2856) 1(15/34) 2(107/238) 1(83/136)
5	Left	1(5/13) 1(6/13) 2(31/52) 1(8/13) 4(2/3)
	Right	4(10/21) 4(13/21) 1(1)
6	Left	1(130/431) 1(185/431) 2(966/2155) 1(202/431) 1(248/431) 2(1319/2155) 1(266/431)
	Right	1(186/529) 1(190/529) 1(197/529) 1(242/529) 1(311/529) 1(312/529) 2(1255/2116) 1(325/529)
7	Left	1(130/431) 1(185/431) 2(966/2155) 1(202/431) 1(248/431) 2(1319/2155) 1(266/431)
	Right	1(455/1528) 1(655/1528) 2(1997/4584) 1(171/382) 1(173/382) 1(179/382) 1(935/1528) 1(943/1528)
8	Left	1(130/431) 1(185/431) 2(966/2155) 1(202/431) 1(248/431) 2(1319/2155) 1(266/431)
	Right	1(120/457) 1(317/914) 2(1136/3199) 1(202/457) 1(517/914) 2(3767/6398) 1(279/457)
9	Left	1(130/431) 1(185/431) 2(966/2155) 1(202/431) 1(248/431) 2(1319/2155) 1(266/431)
	Right	1(76/293) 2(409/1172) 1(106/293) 1(125/293) 2(761/1758) 1(331/586) 1(173/293)
10	Left	1(130/431) 1(185/431) 2(966/2155) 1(202/431) 1(248/431) 2(1319/2155) 1(266/431)
	Right	1(455/1528) 1(655/1528) 2(1997/4584) 1(171/382) 1(173/382) 1(179/382) 1(935/1528) 1(943/1528)
11	Left	1(130/431) 1(185/431) 2(966/2155) 1(202/431) 1(248/431) 2(1319/2155) 1(266/431)
	Right	1(35/117) 1(73/234) 2(407/1170) 2(24/65) 1(53/117) 2(541/936)
12	Left	1(130/431) 1(185/431) 2(966/2155) 1(202/431) 1(248/431) 2(1319/2155) 1(266/431)
	Right	2(58095/212624) 2(14655/53156) 2(19347/53156) 4(117931/265780) 2(120967/212624) 4(125659/212624) 4(649491/1063120)
13	Left	1(130/431) 1(185/431) 2(966/2155) 1(202/431) 1(248/431) 2(1319/2155) 1(266/431)
	Right	1(55104/170581) 1(55118/170581) 2(14614899/43668736) 2(58093535/173310296) 2(15487915/43327574) 2(16180123/43668736) 2(18957635/43668736) 2(19654771/43327574) 2(9904548/21663787) 2(5125179/10917184) 2(102143063/173310296) 4(103517595/174674944) 2(106309919/173310296) 2(26616061/43327574) 2(26959547/43668736)

We tested the algorithm on the counterexamples against the use of resistance spectrums in Fig. 2. The graph indexes are given in Table 2. Using only the primary part of the index was sufficient for distinguishing the non-isomorphic graphs. Therefore, the secondary part of the index is not shown. Next, we generated an index for every simple connected graphs with $\leq$10 vertices. The collisions between the primary parts of the index were observed when the number of vertices reaches eight as shown in Table 3. An example of the collisions is given in Fig. 4. There were a small percentage of graphs that their primary parts of the index collide. Nonetheless, a unique index for every graph was produced once its primary and secondary parts of the index are combined. In practice, the primary part of the index is sufficient for distinguishing the vast majority of graphs. Moreover, the demand for the secondary part of the index appeared to be unnecessary with the decreasing of the graph size.

TABLE 3. Number of collisions between the primary parts of the indexes of all simple connected graphs with $\leq$10 vertices.

#Vertices	#Connected graphs	#Graphs that their indexes collide with the others	
		Using only the primary part of the index	Combining the primary and secondary parts of the index
2	1	0	0
3	2	0	0
4	6	0	0
5	21	0	0
6	112	0	0
7	853	0	0
8	11,117	244 (2.19%)	0
9	261,080	768 (0.29%)	0
10	11,716,571	194,556 (1.66%)	0
Total	11,989,763	195,348 (1.63%)	0

When the number of vertices is eight, there were 122 pairs of non-isomorphic graphs where each pair has the same primary part of the index. When the number of vertices is nine, the collisions between the primary parts of the index increased as observed from 384 graph pairs. When the number of vertices is 10, three and four non-isomorphic graphs with the same primary part of the index were first observed. There were 94,000 graph pairs, 2,148 graph triplets, and 28 graph quadruplets that lead to the collisions. We expected that collisions between the primary parts of the index would occur more frequently with the increasing of the graph size. For instance, collisions among three or four graphs would expand to collisions among any number of graphs. Unfortunately, further experiments were not feasible due to the extremely large number of connected graphs. There are 1,006,700,565 graphs with 11 vertices, and 64,059,830,476 graphs with 12 vertices.

IV. DISCUSSION

We presented a graph indexing algorithm for an exact matching query. A numerical index of a graph can be constructed in polynomial time. The index is composed of primary and secondary parts. The primary part is a list of n fractions where n is the number of vertices. On the other hand, the secondary part is a list of fractions of which its size is variable with the partitioning of vertices. Features of our indexing algorithm are discussed as follows.

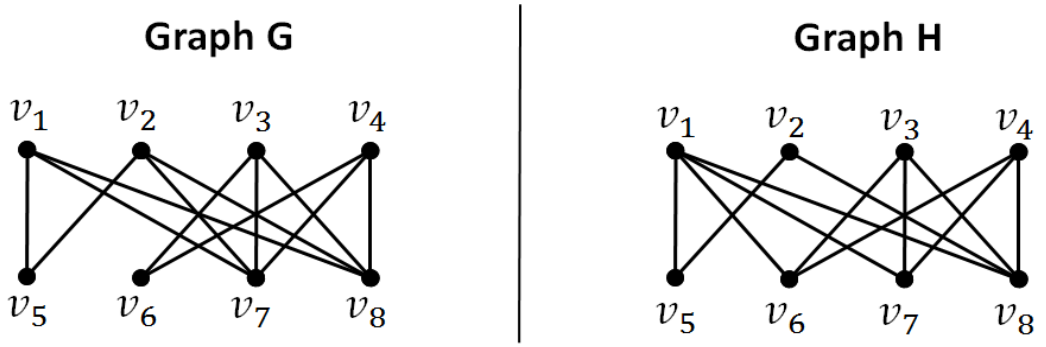


FIGURE 4. Two non-isomorphic graphs have the same primary part of the index. The primary part of the index is (19/65, 19/65, 179/520, 179/520, 179/520, 179/520, 28/65, 28/65). However, their secondary parts of the index are different. The secondary part of the index of graph *G* (left) is (31/80, 31/80, 31/80, 31/80, 31/80, 31/80, 31/80, 31/80, 142/231, 142/231, 142/231, 142/231, 142/231, 142/231, 142/231, 142/231, 65/168, 65/168, 65/168, 65/168, 65/168, 65/168, 65/168, 65/168). On the other hand, the secondary part of the index of graph *H* (right) is (71/159, 74/159, 481/1272, 481/1272, 71/159, 74/159, 71/159, 74/159, 137/321, 149/321, 137/321, 149/321, 86/107, 86/107, 137/321, 149/321, 11/32, 89/96, 11/32, 137/384, 137/384, 89/96, 11/32, 89/96).

- The major difference between our indexing algorithm and the resistance spectrum method is that our algorithm adds a dummy vertex to a graph. The dummy vertex plays an important role for being a reference point during vertex labeling. Every original vertex is labeled by the effective resistance between itself and the dummy vertex. The vertex labels become the primary part of the index and allow the partitioning of vertices. Each block in the partition is perturbed so that the vertices are relabeled to form the secondary part of the index.
- The index is hierarchically separated into primary and secondary parts. Most non-isomorphic graphs can be distinguished by the primary part of the index, which has a fixed length and is easily computed. There were only a few graphs that their non-isomorphisms must be decided by the secondary part of the index, which is length variable and can be computed using more efforts.
- The time complexity of our indexing algorithm is bounded by a polynomial. More precisely, computing an index requires no more than $(n+1)^2$ calculations of the determinant of an $n \times n$ matrix. This is a sharp contrast to the canonical labeling algorithms, which its asymptotic bound is difficult to estimate.
- The calculation of a matrix determinant was efficiently implemented in many software libraries such as LINPACK [26], MATLAB [27], and Mathematica [28]. However, a matrix of Big Fractions was not provided. Thus, we had to implement Big Fraction and used LU decomposition, which is a standard method for calculating a matrix determinant. In the case of sparse matrices, several optimization techniques can speed up the computation of determinants [29]. Moreover, each determinant can be computed independently and extremely fast on a massively parallel computer.
- The performance of determinant computation in our indexing algorithm largely suffers from the lack of primitive operators for Big Fraction. In theory, the time complexity of arithmetic operators for Big Fraction implemented in software grows in a polynomial relationship with the problems size but the execution time in practice is slower than primitive data types by orders of magnitude. This problem can be solved only by implementing arithmetic hardware for Big Fraction.
- We cannot prove that every simple connected graph produces a unique index. If the index is unique, the graph isomorphism problem will immediately be in the complexity class P. The potential of using our indexing algorithm for solving the graph isomorphism problem remains an open problem. On the other hand, a counterexample such as two non-isomorphism graphs producing the same index is also useful for further improvement of our indexing algorithm.
- Our indexing algorithm is not only limited to connected graphs but it is also applicable to disconnected graphs. A disconnected graph can be decomposed into multiple connected subgraphs. The index of each subgraph can be computed individually and can be combined later to form the index of the disconnected graph.

Although some issues mentioned above remain unsolved, our indexing algorithm illustrated an interconnection between graph isomorphism, electrical resistance, and linear algebra. More importantly, our indexing algorithm suggested a polynomial time algorithm for solving the graph isomorphism problem. We sincerely persuade other researchers to prove the conjecture that every simple connected graph produces a unique index or show a counterexample that disproves the conjecture.

REFERENCES

[1] S. Sakr and G. Al-Naymat, "Graph indexing and querying: A review," *Int. J. Web Inf. Syst.*, vol. 6, no. 2, pp. 101–120, 2010.
[2] F. Harary, "The determinant of the adjacency matrix of a graph," *SIAM Rev.*, vol. 4, no. 3, pp. 202–210, 1962.
[3] S. A. Cook, "The complexity of theorem-proving procedures," in *Proc. STOC*, 1971, pp. 151–158.
[4] X. Gao, B. Xiao, D. Tao, and X. Li, "A survey of graph edit distance," *Pattern Anal. Appl.*, vol. 13, no. 1, pp. 113–129, Feb. 2010.

[5] L. Babai and E. M. Luks, "Canonical labeling of graphs," in *Proc. STOC*, 1983, pp. 171–183.
[6] B. D. McKay and A. Piperno, "Practical graph isomorphism, II," *J. Symbolic Comput.*, vol. 60, pp. 94–112, Jan. 2014.
[7] F. Harary, C. King, A. Mowshowitz, and R. C. Read, "Cospectral graphs and digraphs," *Bull. London Math. Soc.*, vol. 3, no. 3, pp. 321–328, 1971.
[8] L. Babai, D. Y. Grigoryev, and D. M. Mount, "Isomorphism of graphs with bounded eigenvalue multiplicity," in *Proc. STOC*, 1982, pp. 310–324.
[9] R. Giugno and D. Shasha, "GraphGrep: A fast and universal method for querying graphs," in *Proc. ICPR*, Aug. 2002, pp. 112–115.
[10] L. Zou, L. Chen, J. X. Yu, and Y. Lu, "A novel spectral coding in a large graph database," in *Proc. EDBT*, 2008, pp. 181–192.
[11] D. W. Williams, J. Huan, and W. Wang, "Graph database indexing using structured graph decomposition," in *Proc. ICDE*, Apr. 2007, pp. 976–985.
[12] H. Jiang, H. Wang, P. S. Yu, and S. Zhou, "GString: A novel approach for efficient search in graph databases," in *Proc. ICDE*, Apr. 2007, pp. 566–575.
[13] D. J. Klein and M. Randić, "Resistance distance," *J. Math. Chem.*, vol. 12, no. 1, pp. 81–95, Dec. 1993.
[14] D. Babić, D. J. Klein, I. Lukovits, S. Nikolić, and N. Trinajstić, "Resistance-distance matrix: A computational algorithm and its application," *Int. J. Quant. Chem.*, vol. 90, no. 1, pp. 166–176, 2002.
[15] L. Sun, W. Wang, J. Zhou, and C. Bu, "Some results on resistance distances and resistance matrices," *Linear Multilinear Algebra*, vol. 63, no. 3, pp. 523–533, 2015.
[16] J. Zhou, Z. Wang, and C. Bu, "On the resistance matrix of a graph," *Electron. J. Combinat.*, vol. 23, no. 1, p. P1.41, 2016.
[17] P. Dimo, *Nodal Analysis of Power System*. Preston, U.K.: Abacus Press, 1975.
[18] E. Estrada and N. Hatano, "Resistance distance, information centrality, node vulnerability and vibrations in complex networks," in *Network Science: Complexity in Nature and Technology*. Springer, 2010, pp. 13–29.
[19] L. Baxter, "Counterexamples Wanted-Graph Isomorphism & Resistances," USENET: sci.math.research, Apr. 22, 1999.
[20] L. Baxter, "Counterexample Wanted for Graph Isomorphism Conjecture," USENET: comp.theory, Apr. 26, 1999.
[21] J. Rickard, "Counterexample Wanted for Graph Isomorphism Conjecture," USENET: comp.theory, Apr.23, 1999.
[22] R. L. Burden and J. D. Faires, *Numerical Analysis*, 9th ed. Boston, MA, USA: Cengage Learning, 2010.
[23] L. Feng, G. Yu, Z. Jiang, and L. Ren, "Sharp upper bounds for the number of spanning trees of a graph," *Appl. Anal. Discrete Math.*, vol. 2, no. 2, pp. 255–259, 2008.
[24] G. R. Grimmett, "An upper bound for the number of spanning trees of a graph," *Discrete Math.*, vol. 16, no. 4, pp. 323–324, 1976.
[25] E. W. Weisstein. (2016). *Resistance-Equivalent Graphs, MathWorld—A Wolfram Web Resource*. [Online]. Available: http://mathworld.wolfram.com/Resistance-EquivalentGraphs.html
[26] J. J. Dongarra, "The LINPACK benchmark: An explanation," in *Proc. ICS*, 1987, pp. 456–474.
[27] *MATLAB 9.0.1*, MathWorks, Natick, MA, USA, 2008.
[28] *Mathematica 10.4*, Wolfram Res., Champaign, IL, USA, 2016.
[29] D. H. Wiedemann, "Solving sparse linear equations over finite fields," *IEEE Trans. Inf. Theory*, vol. 32, no. 1, pp. 54–62, Jan. 1986.

CHATCHAWIT APORNTEWAN received the B.Eng., M.Eng., and D.Eng. degrees from Chulalongkorn University, Bangkok, Thailand, in 1998, 2000, and 2004, respectively, all in computer engineering. He is currently an Associate Professor with the Department of Mathematics and Computer Science, Faculty of Science, Chulalongkorn University, Bangkok, Thailand. He is also with the Omics Sciences and Bioinformatics Center, Chulalongkorn University, Bangkok, Thailand. His research interests include data structures and algorithms for bioinformatics and omics data analysis.

PRABHAS CHONGSTITVATANA (M'10) received the B.Eng. degree in electrical engineering from Kasetsart University, Bangkok, Thailand, in 1980, and the Ph.D. degree in artificial intelligence from the University of Edinburgh, Edinburgh, U.K., in 1992. He is currently a Professor with the Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand. He is a Lifetime Member of the Engineering Institute of Thailand under H. M. The King's Patronage, a Senior Member of the Thai Academy of Science and Technology Foundation, a Senior Adviser of the Thai Robotics Society, and a Founding Member of the Thai Embedded Systems Association. He was the President of the Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology Association of Thailand from 2012 to 2013. He was awarded a National Distinguished Researcher by the National Research Council of Thailand in 2009. He research interests include bioinformatics, computer architecture, evolutionary computation, quantum computing, and robotics.

NACHOL CHAIYARATANA (M'99) received the B.Eng. and Ph.D. degrees in control engineering from the University of Sheffield, Sheffield, U.K., in 1995 and 1999, respectively. He is currently an Associate Professor with the Department of Electrical and Computer Engineering, Faculty of Engineering, King Mongkut's University of Technology North Bangkok, Bangkok, Thailand. His research interests include bioinformatics, evolutionary computation, and machine learning.