Scheduling Voter Checks to Detect Configuration Memory Errors in FPGA-based TMR Systems

Nguyen T. H. Nguyen*, Ediz Cetin†, Oliver Diessel*
*School of Computer Science and Engineering, UNSW Australia
†Department of Engineering, Macquarie University, Australia

***Abstract*—Field-Programmable Gate Arrays (FPGAs) are susceptible to radiation-induced *Single Event Upsets* (SEUs). A common technique for dealing with SEUs is *Triple Modular Redundancy* (TMR) combined with *Module-based configuration memory Error Recovery* (MER). By triplicating components and voting on their outputs, TMR helps localize the configuration memory errors, and by reconfiguring the faulty component, MER swiftly corrects the errors. However, the order in which the voters of TMR components are checked has an inevitable impact on the overall system reliability. In this paper, we outline an approach for computing the reliability of TMR-MER systems that consist of finitely many components. Using the derived reliability models we demonstrate that the reliability of an exemplar system is improved by up to 29% when the critical components are checked more frequently for the presence of configuration memory errors than when they are checked in round-robin order or by up to 11% when the next component to be checked is chosen at run time based on the likelihood that it has failed.**

I. INTRODUCTION

SRAM-based *Field-Programmable Gate Arrays* (FPGAs) are susceptible to radiation-induced *Single Event Upsets* (SEUs). One approach to dealing with SEUs is to use *Triple Modular Redundancy* (TMR) with *Module-based Error Recovery* (MER) [1]–[3]. TMR-MER relies on *Dynamic Partial Reconfiguration* (DPR) to correct configuration memory errors. This recovery method is commonly triggered when repeated errors are detected by the voter(s) associated with a TMR component, and involves rewriting the configuration memory of the module that has been found to be in error. However, the order in which the voters of TMR components are checked has an inevitable impact on the overall system reliability [4]. Moreover, while TMR-MER is generally effective for mitigating SEUs affecting the configuration memory [2], it is not effective at protecting systems against multiple coincident SEUs that affect multiple modules of a TMR component and thus defeat the protection afforded by redundancy.

In this work, we investigate the reliability of TMR-MER systems consisting of any number of triplicated components operating in harsh radiation environments, such as in Geosynchronous Equatorial Orbit (GEO) during solar flares and in high-energy physics laboratories, like the Large Hadron Collider, where multiple coincident SEUs are more probable [5]. Through our research, we hope to show that SRAM-based FPGAs are sufficiently reliable to be used in GEO applications. However, our focus in this paper is in determining the impact on overall system reliability of varying the order and rate at which the voters of TMR components are checked for errors.

II. BACKGROUND AND RELATED WORK

Memory elements in an SRAM-based FPGA device can be classified into two groups: configuration and user memory bits. The configuration memory bits are used to specify the particular circuit mapped into the FPGA, whereas the user memory bits, such as flip-flops or block RAMs, hold the current state of the circuit. The configuration memory bits account for the largest proportion of all the memory cells in SRAM-based FPGAs e.g., more than 80% in the latest Xilinx FPGA (UltraScale XCVU440). Therefore, there is a far greater probability of SEUs occurring in configuration memory bits than in user memory bits. Since the configuration memory upsets have the potential to alter the function of a *look up table* (LUT) or the routing between nodes, they can lead to "permanent" errors manifesting in user circuits until the altered configuration state is corrected. In this work, we study the impact on reliability of multiple SEUs that affect the configuration memory bits in TMR-MER systems.

TMR-MER systems utilize a so-called *Reconfiguration Control Network* (RCN) [2], such as a star-, bus-, or ring-based network, or utilize the in-built *Internal Configuration Access Port* (ICAP) to convey the status bits of the TMR component voters to a *Reconfiguration Controller* (RC), which determines whether configuration memory errors are present and manages the error recovery process. To determine whether any configuration memory upsets have occurred, most TMR-MER systems check the voters of the TMR components in round-robin order [2].

In [4], we developed an on-chip method, known as VSE, for determining the next component to check at run time based on the likelihood that it has failed since the last check. In contrast, this paper reports on an off-line approach to determining a beneficial fixed voter checking sequence (so-called Variable Rate Voter Checking (VRVC)). Our work aims to further enhance the system's error detection capabilities and to thereby raise overall system reliability beyond the improvement possible with VSE. A further benefit of the VRVC approach presented here, which requires simple scheduling code to be added to the RC program, is that it avoids the area overhead and additional design complexity of including the VSE on chip. In [6] we briefly presented simulation results of VRVC and compared these to systems that use round robin for voter checking. In this work, we significantly extend [6] by describing our reliability models for TMR-MER systems and comparing the reliability of an exemplar space-based FPGA system that employs VRVC, VSE and round robin for voter checking.

Reliability models for TMR-MER systems have not yet been studied in detail. When they are mentioned, Markov models are used to compute the system reliability with the assumption that the recovery of modules of multiple TMR components occurs independently [2]. While acceptable at low error rates, the problem with this assumption at high error rates is that the methods for correcting configuration memory errors are inherently sequential, hence the models do not consider the effect of configuration memory errors on other TMR components while a faulty module is being reconfigured.

III. RELIABILITY MODEL

In this section, we introduce models that estimate the reliability of TMR-MER systems. These models are then used to estimate the reliability of FPGA-based designs in harsh

This research was supported in part by the Australian Research Council's Discovery (DP150103866) Project funding scheme.

978-1-5386-0362-8/17/$31.00

radiation environments when multiple coincident upsets are more probable We describe a general reliability model that has been widely used to estimate the reliability of FPGA-based systems. Based on this general model, we outline a procedure for estimating the reliability of TMR-MER systems that consist of an arbitrary number of TMR components and whose voters are checked round-robin order, or at a variable rate.

A. General Reliability Model

The reliability of a TMR component k over time Δt, $R_k(\Delta t)$, can be expressed w.r.t. the component failure probability, $FP_k(\Delta t)$, which is the sum of the individual likelihoods that the component fails for all u SEUs that may affect the device during Δt. These relationships are given in [5] as:

$$\begin{aligned} R_k(\Delta t) &= 1 - FP_k(\Delta t), \\ FP_k(\Delta t) &= \sum_{u=1}^{\infty} P(F_k|E_u)P(E_u, \Delta t), \end{aligned} \tag{1}$$

where event F_k is the failure of component k during the period of time Δt and event E_u is that u SEUs have occurred in the device during the period of time Δt. Failure of TMR component k means that at least two of the three modules suffer from errors and that the component's voter therefore fails to produce the correct output.

$P(F_k|E_u)$ can be estimated for various values of u using the number of sensitive bits per component, for which we use the number of essential bits reported by the vendor's tools as a worst case estimate. Sensitive bits are those bits that cause a functional error if they change state, while essential bits are those bits associated with the circuitry of the design [7].

$P(E_u, \Delta t)$, the probability of event E_u occurring during Δt, can be modelled with a Poisson distribution, $P(E_u, \Delta t) = e^{-\nu}\frac{\nu^u}{u!}$, where ν is the expected number of SEUs suffered by the device during a period of time Δt and is obtained from the product of the failure rate of one configuration memory bit of a device (λ_{bit}), the number of configuration memory bits of a device (n_c) and the time period (Δt): $\nu = \lambda_{bit} \times n_c \times \Delta t$. λ_{bit} depends upon the radiation level, the IC process technology and the circuit architecture of the FPGA fabric.

Once the failure probability of component k is known, the failure rate λ_k of component k is given by [5]:

$$\lambda_k = \frac{FP_k(\Delta t)}{\Delta t}. \tag{2}$$

Since a TMR component can fail in different scenarios (see Fig. 1 and associated discussion in Section III-B) with different failure rates (λ_k^i), it is more meaningful to compute the composite failure rate of each component (λ_k^c). This parameter can be calculated for the expected proportions (ρ_k^i) in which each scenario occurs:

$$\lambda_k^c = \sum_{i=1} \rho_k^i \lambda_k^i. \tag{3}$$

where $\sum \rho_k^i = 1$.

Typically, a system contains N TMR components that are modelled in series from a reliability perspective such that the failure of any one component causes the system to fail. The failure rate of a series TMR system, λ_s, is the sum of all component failure rates [8]. The *Mean Time To Fail* (MTTF) of the system is given by the reciprocal of the system failure rate.

B. Failure rates of TMR-MER systems in which voters are checked in round-robin order

Based on the general reliability model described in Section III-A, we estimate the *failure rate of systems comprised of two TMR components* connected in series. Hereafter, we say that if the output of one module of a TMR component repeatedly differs from that of the other two, that the component is suffering from an "error", and if, after the component suffers another one or more SEUs, the outputs of the remaining two modules repeatedly differ, that the component has "failed". We also assume that once a faulty module is detected, it is dynamically reconfigured to correct the error and to reduce the likelihood of the component failing [2].

TABLE I: Notation

Symbol	Definition
N	Number of TMR components in the system
Ck	Component k, $k = 1..N$
O_{kn}	Ck is observed for the n^{th} time by checking its voter(s)
Δt_o	The time period between successive voter observations (assumed to be constant for a given system setting)
Δt_{dk}	The time period between two consecutive observations of Ck
Δt_{rk}	The time period to recover a faulty module of Ck
Δt_k	The total time period over which Ck can fail
Δt_{dij}	The time period between successive observations of Ci and Cj
$\Delta t_{d'ij}$	The average time period between two consecutive observations of Ci in the interval between two consecutive observations of Cj

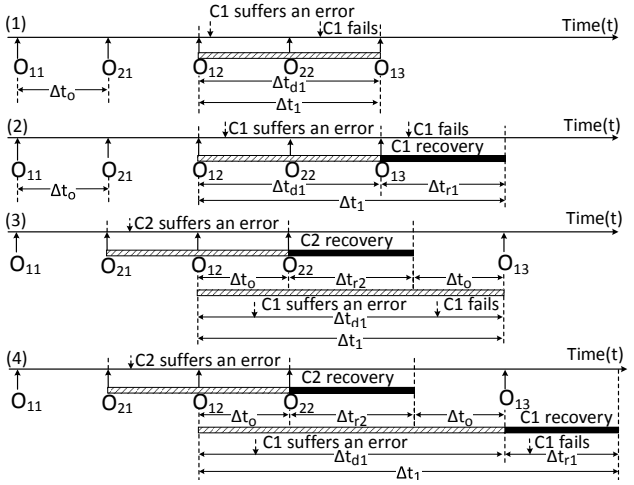

Fig. 1: Failure modes for component 1 in two-component systems in which the voters are checked in round-robin order.

In a two-component system, a component may fail in one of four different ways that are classified into two groups as shown in Fig. 1 and using the notation listed in Table I. Note that Fig. 1 only describes the modes in which C1 can fail; the modes in which C2 can fail can be derived in a similar manner.

Group 0: No other component suffers an error

– Case 1 (Fig. 1(1)): C1 suffers from two or more SEUs that cause it to fail during the period of time between two consecutive checks of its voters (e.g., during Δt_1 – the period of time between O_{12} and O_{13}).

– Case 2 (Fig. 1(2)): C1 suffers an error from one or more SEUs during the period of time between two consecutive checks of its voters (between O_{12} and O_{13} in Fig. 1(2)). Thereafter, C1 fails if one or more SEUs affect its remaining working modules during the period of time that it is recovering from the previous error (e.g., during Δt_{r1} – from time O_{13} to the end of the recovery process of C1).

Group 1: One other component suffers an error

– Case 1 (Fig. 1(3)): C1 suffers from two or more SEUs that cause C1 to fail during a period of time between two consecutive checks of its voters that is longer than usual because the system is recovering from an error in C2. C1 fails during the period

of time that commences after it is observed to be without an error (at O_{12}), continues while C2 is checked and recovered, and finishes when C1 is observed again at O_{13}.

– Case 2 (Fig. 1(4)): C1 suffers an error from one or more SEUs during the period of time between two consecutive checks of it (between O_{12} and O_{13}) while the system is recovering from an error in C2. C1 then fails if one or more SEUs affect a second and/or third module of C1 while it is recovering from the previous error.

To summarize, in case 1 of either group, component k fails, i.e., suffers multiple errors to its different modules, between successive voter checks. In case 2, on the other hand, component k suffers an error to one of its modules during this period, and then fails following subsequent upsets to its other modules while recovering from the first error.

The failure probability of component k in case 1 of either group is computed based on $FP_k(\Delta t)$ in Eq. (1) with corresponding Δt_k as shown in Figs. 1(1) and 1(3).

The failure probability of component k in case 2 of either group is the product of the probability that event M_k (i.e., that component k suffers an error) occurs during the period of time Δt_{dk} as shown in Figs. 1(2) and 1(4) and that component k fails during the period of time Δt_{rk} given the occurrence of event M_k.

Based on Eq. (2), the failure rate of component k (λ_k^i) in each case is estimated using the corresponding Δt_k (Fig. 1).

The proportions ρ_k^i are calculated for the likelihood by which component k fails in each case. For example, the likelihood of cases in group 0 occurring depends upon the likelihood that component k suffers an error, while that of cases of group 1 occurring depends upon the likelihood that both components suffer an error.

The composite failure rate of component k (λ_k^c) is calculated by substituting λ_k^i and ρ_k^i into Eq. (3), and the system failure rate can be computed by summing λ_k^c for all k.

The reliability of *systems comprising any number of TMR components* can be readily computed by extending the approach we have outlined for two-component systems by considering all possible cases in which each component may fail [9].

C. Failure rates of TMR-MER systems employing VRVC

Variable-Rate Voter Checking (VRVC) is defined as a periodic schedule in which component voters are checked at specific times and in which the more vulnerable components' voters are checked more frequently than those of the less vulnerable ones. For example in a system of 4 components, one period of a schedule could be *4-3-4-2-4-3-4-2-3-1* in which each digit represents the component whose voters are to be checked. In this case, component 4 is deemed more vulnerable and hence checked more frequently when compared to the other components, and component 1 is deemed least vulnerable and hence checked less frequently.

1) A 2-component system: Similar to the cases described in Section III-B, we observe that C1 fails in one of four different ways as partly depicted in Fig. 2 using the notation of Table I. Note that p in Fig. 2 denotes the nominal number of times that C2 is checked between two consecutive checks of C1 due to its greater susceptibility to SEUs than C1's. In case 1 of group 1 (Fig. 2(3)), we assume that the system detects an error in C2 x checks after C1 is checked (at O_{12}) where x varies from 1 to p. In this work, we associate with $x = 1..p$ the number of checks that the system performs before it detects an error in C2. Thus, each case of group 1 involves p sub-cases that have the same likelihood of both components suffering an error (ρ_k^i). For example, given a schedule of two components in the following order *1-2-2-2-2-1-2-2-2-2-1...* where each digit

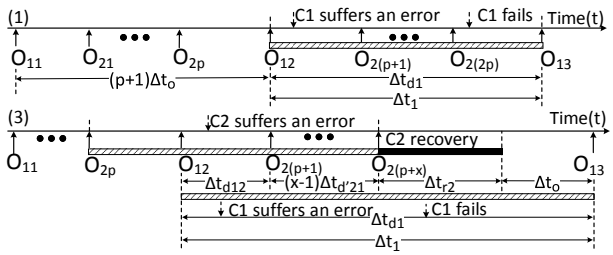

Fig. 2: Failure modes for component 1 in systems comprising two components, which employs variable-rate voter checking

denotes the observation of the corresponding component, Δt_{d1} and Δt_{d2} in group 0 are $5\Delta t_o$ and $1.25\Delta t_o$, respectively. Both Δt_{d12} and $\Delta t_{d'21}$ in group 1 are Δt_o. Furthermore, with such a schedule, there are four checks of C2 during the period of time between two consecutive checks of C1. Thus, $x = 1..4$.

The observations of C2 differ slightly from those of C1. C2 may also fail in one of four different ways, but the number of sub-cases in group 1 is only 1. This is because between any two consecutive checks of C2, C1 is checked at most once.

The above observations allow us to compute the system failure rate.

2) An N-component system: We assume that the components are numbered $k = 1..N$ and ranked into non-decreasing vulnerability order, and that component k is therefore not checked less frequently than component k–1. After the reconfiguration of a faulty module is finished, the system checks all other components in descending order of vulnerability before recommencing the planned schedule.

The system failure rate can be computed by considering all possible cases in which each component may fail [9].

IV. SCHEDULING VOTER CHECKS

We surmise that the problem of statically *determining the optimal number of voter checks per period* in an N-component system is NP-hard. We therefore propose a *genetic algorithm* (GA), which is a probabilistic search method based on an evolutionary approach, to heuristically determine the rate at which all triplicated components in a system should be checked so as to maximize the system reliability. Once the rate at which components should be checked has been determined, we use a second GA, as detailed in [10], to *generate a schedule in which the determined number of voter checks are evenly distributed over a schedule period*. The schedule produced by the second GA is used to evaluate the fitness of individual solutions to the first GA, which determines the number of checks to be performed per period.

V. EXPERIMENTAL ANALYSIS

In this work, we evaluate and compare the performance of the VRVC approach with that of VSE and round-robin voter checking when each are implemented on an experimental CubeSat payload known as RUSH [2] operating in GEO.

A. Experiments

The RUSH payload consists of the 9 TMR components listed in Table II hosted on an Artix-7 XC7A200TFBG-484 FPGA from Xilinx. These components are representative of circuits that are commonly included in space-based applications and that utilize a mixture of FPGA resources. They include: a single MAC-based 21-tap *Finite Impulse Response* (FIR) filter with 16-bit signal width; an 8-to-3-bit *Block Adaptive Quantizer* (BAQ); an 8,096-word deep 32-bit FIFO; three 32-bit *Shift Registers* (SRs) having different lengths and a variety

TABLE II: Results of 9 TMR components

Design	Ess. bits n_e	RC t_r (ms) – # checks (d_k)			
		100MHz	50MHz	20MHz	10MHz
BST3	1,833,235	26.7 – 47	49.5 – 45	72.4 – 47	118.7 – 49
SR3	1,403,647	19.6 – 41	43.8 – 40	64.0 – 39	104.9 – 46
BST2	793,534	11.0 – 28	24.5 – 31	35.8 – 34	58.7 – 36
SR2	515,904	8.5 – 27	21.7 – 29	31.7 – 33	52.0 – 29
SR1	285,914	6.8 – 26	13.6 – 24	19.9 – 25	32.6 – 25
BST1	281,604	2.6 – 23	5.9 – 23	8.6 – 20	14.0 – 25
BAQ	48,963	1.3 – 15	3.0 – 18	4.4 – 18	7.1 – 14
FIFO	41,842	3.5 – 12	7.8 – 12	11.4 – 13	18.7 – 13
FIR	12,042	1.2 – 08	2.6 – 11	3.9 – 10	6.3 – 11

of combinational functions between the stages; and three 32-bit *Binary Search Trees* (BSTs) of different heights and a variety of combinational functions at each node. The designs were implemented using Vivado 2014.4 with default settings. Due to power limitations of the CubeSat that deploys the exemplar system, all components are operated at 10MHz. We have evaluated system MTTF using GEO worst-day radiation conditions with a bit error rate of 7.34E-11 upset/bit/s [4] to assess the potential to use the system under more extreme radiation conditions.

An RC using the ICAP-based voter checking approach [2] is used to read the voter status bits. The RC includes a MicroBlaze (MB) processor connected to an *External Memory Controller* (EMC), a DMA *Controller* (DMAC) and the Xilinx AXI HWICAP IP accessed via an AXI bus. The MB processor configuration is created with minimal features and can be operated at 100MHz, 50MHz, 20MHz, or 10MHz. The AXI HWICAP IP combines with EMC and DMAC to reconfigure faulty modules and is also used for flipping configuration memory bits during the fault injection experiment.

We performed a fault injection experiment to assess the *Mean Time To Detect* (MTTD) errors in the RUSH system using each of the three voter checking schedules. The RC receives a random configuration bit address generated by a host PC, reads the corresponding frame, flips the addressed bit and writes the frame back using the HWICAP to emulate the occurrence of a memory error.

B. Results and discussions

Table II reports the number of essential bits (n_e) and the recovery times (t_r) per triplicated module, which is the time interval between an error being detected in a module until the last word of the partial bitstream used to recover that module is written back to the FPGA via the HWICAP. The table also reports the number of checks (d_k) made of each component per VRVC schedule period to achieve the reported MTTF.

TABLE III: MTTF, MTTD and power consumption at various RC clock frequencies in GEO.

RC operating frequency		**100MHz**	**50MHz**	**20MHz**	**10MHz**
Voter observ. period Δt_o		**71μs**	**142μs**	**355μs**	**711μs**
MTTF (years)	Round-robin	103.0(-15%)	49.0(-15%)	28.0(-20%)	16.0(-23%)
	VSE	116.4(-4%)	54.9(-5%)	32.6(-7%)	19.2(-7%)
	VRVC	121.7(0%)	57.6(0%)	35.1(0%)	20.7(0%)
MTTD (μs)	Round robin	320(-39%)	639(-37%)	1596(-45%)	3200(-53%)
	VSE	290(-26%)	580(-25%)	1451(-31%)	2905(-39%)
	VRVC	230(0%)	465(0%)	1105(0%)	2088(0%)
Power (mW)	RC	252(0%)	196(-22%)	163(-35%)	152(-40%)
	RC+TMR comp.	456(0%)	394(-14%)	357(-22%)	344(-25%)

Table III reports three metrics. The first is the MTTF in years (and percentage MTTF decrease) for the implementations employing round robin, VSE and VRVC (and w.r.t the VRVC system) for voter checking in GEO. The second is the MTTD errors using the round-robin, VSE and VRVC approaches. The third is the power consumption in mW of (i) the RC on its own, and (ii) the RC including the 9 components, when the RC is operated at different clock frequencies. The percentage reduction in power consumption, relative to the RC operating at 10MHz, is indicated in parentheses. This power consumption figure relates to the energy expended checking the voters at intervals of Δt_o, and therefore applies to all schedules equally.

We found that the TMR-MER system using VRVC is more reliable than the same system using round robin when the available power in the system is constrained. Table III shows that the system reliabilities (as given by the MTTF) are proportional to the rates at which the system recovers from errors (t_r from Table II). However, for the sake of saving energy in space-based applications during long missions, the voter checking frequency can be significantly reduced [11]. For example, when the RC runs at 10MHz compared to 100MHz, the energy consumption of the RC alone is reduced by 40% and that of the whole system is reduced by 25% (Table III). In this case, the ratio of the MTTF achieved using VRVC to that obtained using round robin for voter checking increases from 118% for the RC operating at 100MHz to 129% at 10MHz. Importantly, the expected MTTF of 21 years exceeds the expected lifetime of GEO satellites. It can also be observed that the MTTFs of systems employing VRVC are greater than those that employ VSE at all four RC clock frequencies.

Table III also shows that VRVC allows errors to be detected 44% faster on average than with round robin and 30% faster than when VSE is used to check voters.

VI. CONCLUDING REMARKS

We have presented reliability models for TMR-MER systems having a finite number of components and whose voters are checked in round-robin order and at variable rates. We assert that any FPGA-based TMR system which uses a reconfiguration control network that provides random access to component voters can benefit from variable-rate scheduling to prioritize checks of more vulnerable components. The benefits become more significant as the radiation level increases and/or as the checking frequency decreases.

REFERENCES

[1] C. Bolchini et al., "A novel design methodology for implementing reliability-aware systems on SRAM-based FPGAs," *IEEE Trans. on Computers*, vol. 60, 2011.

[2] D. Agiakatsikas et al., "Reconfiguration control networks for TMR systems with module-based recovery," in *FCCM*, May 2016.

[3] M. Straka et al., "Fault tolerant system design and SEU injection based testing," *Microprocessors and Microsystems, vol. 37*, 2013.

[4] N. Nguyen et al., "Dynamic scheduling of voter checks in FPGA-based TMR systems," in *FPT*, Dec 2016.

[5] P. Ostler et al., "SRAM FPGA reliability analysis for harsh radiation environments," *IEEE Trans. on Nuclear Science*, vol. 56, Dec 2009.

[6] N. Nguyen et al., "Scheduling Considerations for Voter Checking in TMR-MER Systems," in *FCCM*, May 2017.

[7] R. Le, "Soft error mitigation using prioritized essential bits," *Xilinx XAPP538 (v1.0)*, 2012.

[8] I. Koren and C. M. Krishna, *Fault-Tolerant Systems*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007.

[9] N. Nguyen et al., "Scheduling Considerations for Voter Checking in FPGA-based TMR Systems," UNSW CSE, Tech. Rep., May 2017.

[10] A. Garca-Villoria et al., "Solving the response time variability problem by means of a genetic algorithm," *European Journal of Operational Research*, vol. 202, 2010.

[11] I. Herrera-Alzu et al., "Design techniques for Xilinx Virtex FPGA configuration memory scrubbers," *IEEE Trans. on Nuclear Science*, vol. 60, 2013.