# PALBBD: A Parallel ArcLength Method Using Bordered Block Diagonal Form for DC Analysis

Zhou Jin[1], Tian Feng[1], Yiru Duan[1], Xiao Wu[2], Minghou Cheng[2], Zhenya Zhou[2], Weifeng Liu[1]
1. Super Scientific Software Laboratory,
Department of Computer Science and Technology, China University of Petroleum-Beijing, Beijing, China
2. Huada Empyrean Software Co. Ltd, Beijing, China
Email: jinzhou@cup.edu.cn, fengtian_leo@outlook.com, yiruduan99@163.com, wuxiao@mail.empyrean.com.cn, chengmh@mail.empyrean.com.cn, zhouzhy@mail.empyrean.com.cn, weifeng.liu@cup.edu.cn

## ABSTRACT

With the increasing complexity of integrated circuits, it is becoming cumulatively challenging to solve the entire large-scale nonlinear algebraic system in DC analysis within reasonable simulation time and without accuracy lost. For this reason, we present an efficient parallel arclength approach called PALBBD to solve DC problems for large capacity and full accuracy in this paper. We process the $m$+1 dimensions equation of the Newton-Raphson (NR) iteration in an alternative way, which maintains the Jacobian matrix structure. Besides, we exploit the bordered block diagonal (BBD) form to save the matrix for parallel computing. Moreover, we check the convergence of each sub-partition and bypass the calculations of converged ones to reduce the amount of unnecessary computations during the iteration. In order to ensure the accuracy, we use a correction equation to replace the Schur complement updating for the bypassed sub-partitions. The proposed PALBBD is implemented and integrated to the SPICE simulator and verified by 72 real-world circuits. It outperforms the conventional serial arclength method with up to 73.93X speedup and 45% bypass ratio.

## CCS CONCEPTS

• **Hardware → Software tools for EDA**.

## KEYWORDS

Parallel DC analysis, circuit simulation, arclength, bypassing, bordered block diagonal

**ACM Reference Format:**
Zhou Jin[1], Tian Feng[1], Yiru Duan[1], Xiao Wu[2], Minghou Cheng[2], Zhenya Zhou[2], Weifeng Liu[1]. 2021. PALBBD: A Parallel ArcLength Method Using Bordered Block Diagonal Form for DC Analysis. In *Proceedings of the Great Lakes Symposium on VLSI 2021 (GLSVLSI '21), June 22–25, 2021, Virtual Event, USA*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3453688.3461526


*GLSVLSI '21, June 22–25, 2021, Virtual Event, USA.*

ACM ISBN 978-1-4503-8393-6/21/06…$15.00
https://doi.org/10.1145/3453688.3461526

## 1 INTRODUCTION

SPICE-like circuit simulator is the most widely used computer-aided tool for circuit designs[2]. It provides several analysis types to evaluate designer's circuits from various viewpoints before tape out. Among them, computing DC operating points is the most essential task, since it is performed prior to any other simulation mode. It provides initial solution for transient analysis and determines linearized, small signal models for all nonlinear devices in AC analysis[2–4]. With the rapid progress of VLSI (Very Large Scale Integration) circuits, the feature size is greatly scaling down and the amount of devices can easily reach several millions especially after post-layout extraction[7]. It makes SPICE-like simulators quite time-consuming to compute DC operating points. Thus, parallel DC analysis which maintains the SPICE accuracy is becoming a critical part of circuit simulators for handling large-scale circuits.

However, there are three main challenges. Firstly, though several parallel techniques have been proposed for circuit simulation in some literatures, the strategies mainly focus on parallel model evaluation[12] or solving the linear system[5–7, 12, 14, 15]. How to solve the DC problem efficiently in parallel is a major challenge. Secondly, determining DC operating points needs to solve a series of nonlinear algebraic equations. The arclength method is an effective approach to trace the solution curve, and in general has a strong ability to eliminate the non-convergence problem compared with other continuation methods[10]. However, it needs to enlarge the Jacobian matrix structure. How to provide a good interface to the existing software is a major problem. Thirdly, as the number of calculations required continues to increase, the computational pressure becomes greater, and how to remove unnecessary calculations has become an important challenge.

In response to the three difficulties mentioned above, we propose the following solutions. Firstly, a parallel arclength approach utilizing the bordered block diagonal form matrix (PALBBD) is presented for accurate and fast DC analysis. Secondly, the proposed method processes the $m$+1 dimensions equation of the NR iteration in an alternative way, which does not need to enlarge the Jacobian matrix structure. Thus the proposed method can be easily integrated into any current existing SPICE-like simulators without changing the data structure. Thirdly, we propose a new bypassing strategy for each sub-partition to minimize the amount of computation. The new bypassing strategy is proposed with a correction equation to replace the top-layer's Schur complement and ensure the accuracy. We proved the lossless accuracy. Different from all conventional bypassing approaches, substitution for converged sub-partitions and even part device evaluation for not-converged sub-partitions

could also be bypassed in our approach. Besides, an initial solution method is used for better convergence.

In this paper, we mainly have the following three contributions: (1) We present a fast parallel arclength approach for accurate and fast DC analysis, (2) The proposed method provides a simple interface and is easy to be integrated to SPICE simulators, (3) The proposed bypassing strategy decreases unnecessary computations without accuracy loss. Experimental results on 72 real-world large-scale circuits show that the proposed PALBBD method achieves 10.2X speedup on 16 threads over conventional serial method on average and up to 73.93X maximum. The proposed bypassing method further provides up to 45% reduction of the computation.

## 2 PRELIMINARY

### 2.1 Arclength Method

Finding DC operating points needs to solve a set of nonlinear algebraic equations as shown in (1) that describes the DC behavior of a nonlinear circuit.

$$F(\boldsymbol{x}) = 0, \boldsymbol{x} \in \boldsymbol{R^m}, F(\boldsymbol{x}) \in \boldsymbol{R^m} \rightarrow \boldsymbol{R^m}, \tag{1}$$

where $\boldsymbol{x}$ is the unknown vector of node voltages and internal currents of independent voltage sources, and $\boldsymbol{m}$ is the number of unknowns. The fixed point homotopy method[3, 4, 13] is globally convergent and in general has a strong ability to achieve bifurcation free with a simple form as shown in (2), in which $\boldsymbol{F(x)}$ is the original system to solve, $\boldsymbol{G}$ is an incidence matrix with constant entries 1e-3 in the diagonal position.

$$\boldsymbol{H}(\boldsymbol{x}, \lambda) = \lambda \boldsymbol{F}(\boldsymbol{x}) + (1-\lambda)\boldsymbol{G} \cdot \boldsymbol{x} = 0. \tag{2}$$

There are a number of methods for tracing the solution-curve of homotopy functions[10, 11]. The arclength is considered as one of the most efficient algorithms, since it can trace the solution-curve continuously[10]. It considers $\lambda$ as a one-dimensional unknown variable and all the variables are regarded as a function of $s$, where $s$ satisfies

$$\sum_{i=1}^{m} (\frac{dx_i}{ds})^2 + (\frac{d\lambda}{ds})^2 = 1. \tag{3}$$

The $\lambda$ is solved at each time point with the nonlinear system together.

### 2.2 BBD Form

BBD is an efficient method for solving linear system in parallel[8, 14]. Apply hypergraph partitioning to the circuit to yield several non-overlapping sub-partitions. Sub-partitions are stored in the diagonal bordered blocks in the matrix as shown in Fig. 1 and can be computed completely in parallel. Also, a top partition is constructed which contains the adjacency node information with all other sub-partitions. For each sub-partition, the Schur complement for top matrix is formed by equation (4), where $\dot{\mathbf{TOP}}_{\_B_i}$ is the corresponding Schur complement of $B_i$ for top matrix.

$$\begin{aligned} &\begin{bmatrix} \mathbf{B}_i & \mathbf{C}_i \\ 0 & \dot{\mathbf{TOP}}_{\_B_i} \end{bmatrix} \begin{bmatrix} \Delta \mathbf{X}_{B_i} \\ \Delta \mathbf{X}_{TOP_{B_i}} \end{bmatrix} = \begin{bmatrix} \mathbf{RHS}\_B_i \\ \dot{\mathbf{RHS}}_{TOP\_B_i} \end{bmatrix} \\ &\dot{\mathbf{TOP}}_{\_B_i} = \mathbf{TOP}_{\_B_i} - \mathbf{D}_i \cdot \mathbf{B}_i^{-1} \cdot \mathbf{C}_i \\ &\dot{\mathbf{RHS}}_{TOP\_B_i} = \mathbf{RHS1}_{TOP\_B_i} - \mathbf{D}_i \cdot \mathbf{B}_i^{-1} \cdot \mathbf{C}_i. \end{aligned} \tag{4}$$

After accumulating the Schur complement, the top level partition can be solved firstly, and then each sub-partition can be solved.

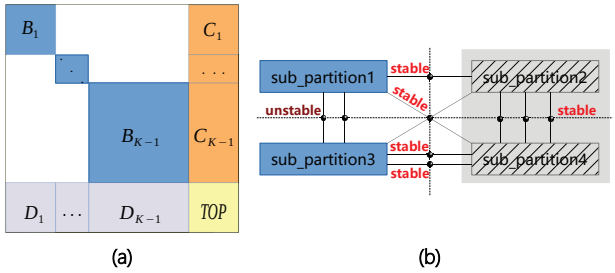


**Figure 1: (a) The circuit is partitioned into several sub-partitions and stored as a BBD form matrix. (b) When internal nodes and external nodes of some sub-partitions are stable, these partitions are regarded as convergent and do not need to be updated and calculated. The NR iteration of these partitions can be bypassed in PALBBD, and the remaining not-convergent sub-partitions will continue to iterate until all of them are converged.**

## 3 PALBBD METHOD

### 3.1 Parallel Arclength Approach

Considering homotopy function in (2), at each arclength iteration step, the tangent vector $\boldsymbol{v} = [dx_1/ds, dx_2/ds, ...d\lambda/ds]$ can be calculated by solving a system of linear equations.

$$\begin{bmatrix} (1-\lambda)\boldsymbol{G} + \lambda \cdot \boldsymbol{J}(\boldsymbol{x}) & -\boldsymbol{G} \cdot \boldsymbol{x} + \boldsymbol{F}(\boldsymbol{x}) \\ (\boldsymbol{v}_x^p)^T & v_\lambda^p \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_x \\ v_\lambda \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix}. \tag{5}$$

In order not to enlarge the matrix size, our PALBBD computes $v_\lambda$ firstly through (6).

$$v_\lambda = \left\{ v_\lambda^p - (\boldsymbol{v}_x^p)^T [\lambda \boldsymbol{J}(\boldsymbol{x}) + (1-\lambda)\boldsymbol{G}]^{-1} \cdot [\boldsymbol{F}(\boldsymbol{x}) - \boldsymbol{G} \cdot \boldsymbol{x}] \right\}^{-1}, \tag{6}$$

and then we can obtain the vector $\boldsymbol{v}_x$ from

$$\boldsymbol{v}_x = v_\lambda \cdot [\lambda \boldsymbol{J}(\boldsymbol{x}) + (1-\lambda)\boldsymbol{G}]^{-1} \cdot [\boldsymbol{G} \cdot \boldsymbol{x} - \boldsymbol{F}(\boldsymbol{x})]. \tag{7}$$

With the normalization of vector $\boldsymbol{v}_\lambda$ and $\boldsymbol{v}_x$, the unit tangent vector at current time step can be obtained as $\boldsymbol{v}^k$ at $k$th arclength iteration. If we set $h$ as the step size, the next time point's predictor $\tilde{\boldsymbol{x}}^{k+1}$ is determined by:

$$\tilde{\boldsymbol{x}}^{k+1} = \boldsymbol{x}^k + \boldsymbol{v}^k \cdot h. \tag{8}$$

Then in the corrector step, we need to solve nonlinear equations in (9) by the NR method to obtain the intersection of the hyperplane, that is perpendicular to the tangent vector $\boldsymbol{v}$ and through the predictor $\tilde{\boldsymbol{x}}^{k+1}$, and the curve $\boldsymbol{H}(\boldsymbol{x}) = \mathbf{0}$.

$$\begin{aligned} &\boldsymbol{H}(\boldsymbol{x}, \lambda) = \lambda \boldsymbol{F}(\boldsymbol{x}) + (1-\lambda)\boldsymbol{G} \cdot \boldsymbol{x} = \mathbf{0} \\ &\boldsymbol{G}(\boldsymbol{x}, \lambda) = \boldsymbol{v}_x^T \cdot (\boldsymbol{x} - \tilde{\boldsymbol{x}}^{k+1}) + v_\lambda \cdot (\lambda - \tilde{\lambda}^{k+1}) = \mathbf{0}. \end{aligned} \tag{9}$$

Thus at each arclength step, the NR iteration can be written as:

$$\begin{bmatrix} \boldsymbol{J}(\boldsymbol{x}) + (1-\lambda) \cdot \lambda^{-1} \cdot \boldsymbol{G} & (\boldsymbol{F}(\boldsymbol{x}) - \boldsymbol{G} \cdot \boldsymbol{x}) \cdot \lambda^{-1} \\ \boldsymbol{v}_x^T & v_\lambda \end{bmatrix} \begin{bmatrix} d\boldsymbol{x} \\ d\lambda \end{bmatrix}$$

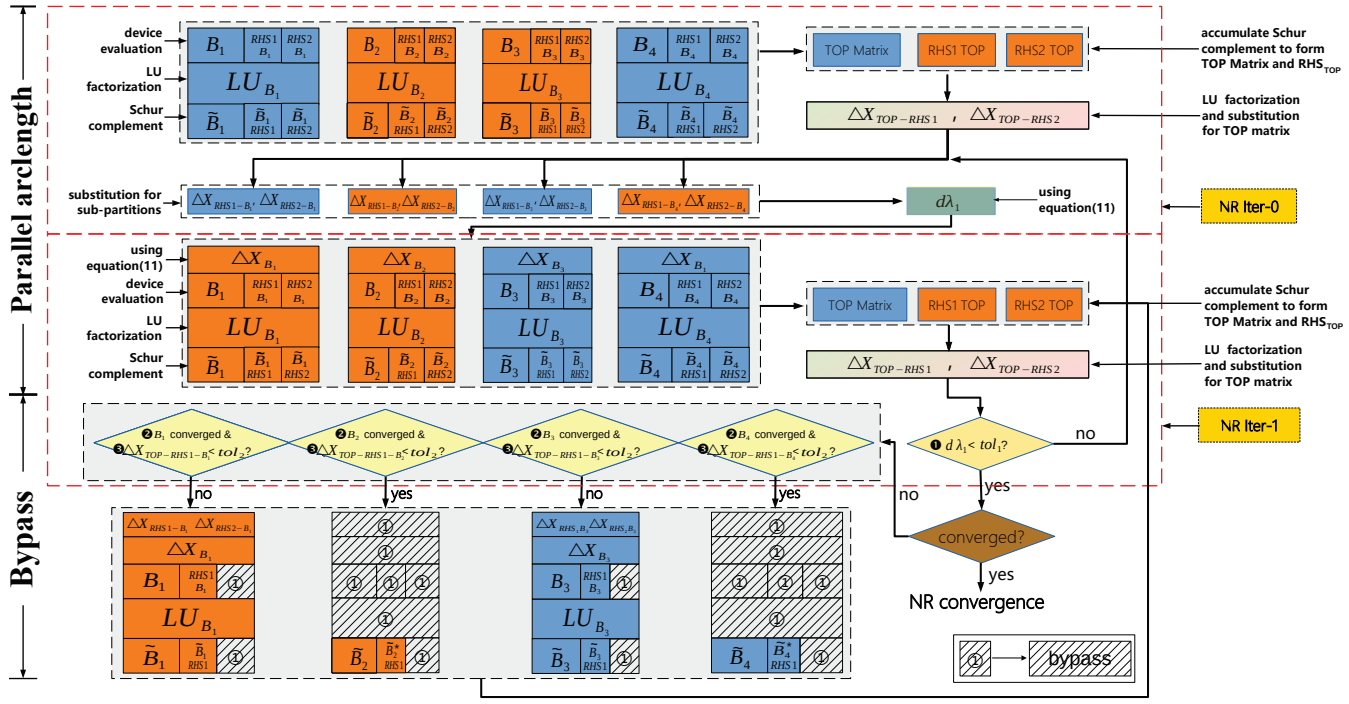


**Figure 2: The proposed PALBBD approach at each arclength step. Firstly, for each sub-partition, we have three sub-tasks: device evaluation to form matrix and RHS in (12), the LU factorization and computing the Schur complement. Three sub-tasks are merged as one for each sub-partition and can be done in parallel. Here the same color is used to indicate tasks on the same processor as an example. After that, the Schur complements are accumulated to form the top-level matrix and RHS. Then we solve the top-level matrix to obtain the solutions of external nodes and send them back to each sub-partition to obtain the internal nodes solutions. Here, all the solutions mean the solutions of (12). Then the $d\lambda$ can be computed by (11). Thus external and internal solutions of (10) for each sub-partition can be obtained by (11) in parallel. Do the loop and from the second NR iteration, three bypassing conditions are checked for each sub-partition. If the $d\lambda$ is converged, internal solutions of a certain sub-partition are converged, and its external solutions from top matrix equation are also converged, then this sub-partition can be regarded as converged. Converged sub-partitions can be bypassed in the next iteration, such as B2 and B4 in the figure. Different from conventional bypassing methods, substitution for each converged sub-partition and computations related to RHS2 for not-converged sub-partitions can also be bypassed. Besides, a corrector $B_i^*$ is exploited to form the RHS for top-level matrix to maintain the accuracy. After bypassing, we redistribute the load for the rest of the work until final converge.**

$$= \begin{bmatrix} -\boldsymbol{F}(\boldsymbol{x}) - (1-\lambda)\cdot\lambda^{-1}\cdot\boldsymbol{G}\cdot\boldsymbol{x} \\ -\boldsymbol{v}_x^T(\boldsymbol{x}-\tilde{\boldsymbol{x}}^{k+1}) - v_\lambda(\lambda-\tilde{\lambda}^{k+1}) \end{bmatrix}. \tag{10}$$

To construct the matrix in (10) needs to enlarge the Jacobian matrix to $m$+1 dimensions. In PALBBD, we conceptually save this matrix as a BBD structure. Thus, we can obtain

$$\begin{cases} d\lambda = \left[-\boldsymbol{v}_x^T(\boldsymbol{x}-\tilde{\boldsymbol{x}}^{k+1}) - v_\lambda(\lambda-\tilde{\lambda}^{k+1}) - \boldsymbol{v}_x^T\boldsymbol{A}^{-1}\cdot\mathbf{RHS}_1\right] \\ \quad\cdot\left(v_\lambda - \boldsymbol{v}_x^T\cdot\boldsymbol{A}^{-1}\cdot\mathbf{RHS}_2\right)^{-1} \\ d\boldsymbol{x} = \boldsymbol{A}^{-1}\cdot\mathbf{RHS}_1 - d\lambda\cdot\boldsymbol{A}^{-1}\cdot\mathbf{RHS}_2 \\ \mathbf{RHS}_1 = -\boldsymbol{F}(\boldsymbol{x}) - (1-\lambda)\cdot\lambda^{-1}\cdot\boldsymbol{G}\cdot\boldsymbol{x} \\ \mathbf{RHS}_2 = [\boldsymbol{F}(\boldsymbol{x}) - \boldsymbol{G}\cdot\boldsymbol{x}]\cdot\lambda^{-1} \\ \boldsymbol{A} = \boldsymbol{J}(\boldsymbol{x}) + (1-\lambda)\cdot\lambda^{-1}\cdot\boldsymbol{G}. \end{cases} \tag{11}$$

Therefore, in our PALBBD, firstly we solve the linear systems in (12), where $\boldsymbol{A}$ is a BBD form matrix as shown in Fig.1(a), and the solution of (12) is exactly the value of $\boldsymbol{A}^{-1}\cdot\boldsymbol{RHS}_1$ and $\boldsymbol{A}^{-1}\cdot\boldsymbol{RHS}_2$.

$$\mathbf{A} * \begin{bmatrix} \Delta\mathbf{X}_{RHS_1} & \Delta\mathbf{X}_{RHS_2} \end{bmatrix} = \begin{bmatrix} \mathbf{RHS}_1 & \mathbf{RHS}_2 \end{bmatrix}. \tag{12}$$

Each sub-partition can be processed in parallel as shown in Fig. 2. The top matrix does not contain any devices, and the Schur complements from each sub-partition are accumulated to form the top Jacobian matrix and two RHS, $\mathbf{RHS}_{1\_TOP}$ and $\mathbf{RHS}_{2\_TOP}$. Through the LU factorization and substitution of the top matrix, we can get the solutions $\Delta\boldsymbol{X}_{TOP\_RHS1}$ and $\Delta\boldsymbol{X}_{TOP\_RHS2}$ of the top matrix for each linear system. Then the solutions for each sub-partition can be also obtained. Therefore, at each NR step, after solving two sets of linear systems, through (11), $d\lambda$ can be obtained, and then we can get $d\boldsymbol{x}$ for each sub-partition as $\Delta\boldsymbol{X}_{B_i}$. This process can be considered as step 0 of the NR iteration.

In the proposed scheme, we merge the device evaluation, LU factorization, Schur complement computation and substitution for each sub-partition as one task. Thus, only one synchronization

is needed and a better load balance can be achieved due to less synchronization and less waiting. It is easy to find that the data structure of $\boldsymbol{J}(\boldsymbol{x})$ and RHS are maintained in PALBBD, and the size of Jacobian matrix preserves as $m \times m$. Because of this, the arclength method can be easily integrated into the SPICE-like simulator.

Besides, in order to avoid the time step size to be reduced at the first arclength iteration step, we fix the value of $\lambda$ at the first step instead of solving it together with the whole nonlinear system to make it easier to converge for the NR iteration. That is, the nonlinear equation H(x,$\lambda$) with $m$ unknowns is solved with a given $\lambda$. Since there is no prior tangent vector $\boldsymbol{v}$, an initial $\boldsymbol{v}_x^0$ and $v_\lambda^0$ is set as [0,1] in (5) to help guess the unit tangent vector at the first step.

## 3.2 Bypassing Process

In this section, we propose a bypassing strategy for sub-partitions to significantly reduce the time-consuming computational tasks during iteration. Moreover, we propose a correction equation for Schur complement formulation to maintain the accuracy. In order to minimize the computation as much as possible, from the second NR iteration, we verify the following three conditions for each sub-partition before the substitution.

- **Condition 1**: If $d\lambda$ at previous time step is small enough, that is $d\lambda < tol_1$, where $tol_1$ is a given tolerance.
- **Condition 2**: If for any sub-partition, the solutions of internal nodes are converged, that is $\Delta \boldsymbol{X}_{B_i} < tol_2$ and $\boldsymbol{RHS}_{1\_B_i} < tol_2$, where $tol_2$ is a given tolerance.
- **Condition 3**: If for any sub-partition, the solutions of external nodes are converged, that is $\Delta \boldsymbol{X}_{TOP\_RHS1\_B_i} < tol_3$, where $\Delta \boldsymbol{X}_{TOP\_RHS1\_B_i}$ is the solution of top matrix that related to sub-partition $B_i$ and $tol_3$ is a given tolerance.

From (10), we can find $\boldsymbol{RHS}_1$ is exactly the RHS of the original homotopy function that corresponding to the $d\boldsymbol{x}$. As if three conditions are satisfied for one sub-partition, the internal and external nodes and the globally parameter *lambda* are all converged. Then this sub-partition can be regarded as converged and can be treated as a large linear device, as shown in Fig. 1(b). Therefore, substitution, device evaluation, LU factorization and Schur complement computation can be bypassed for this sub-partition. Next, only the other not-converged sub-partitions need to be solved at the next step iteration. Moreover, for these not-converged sub-partitions, computations related to $\boldsymbol{RHS}_2$ could also be bypassed according to (11) since $d\lambda$ tends to be 0.

Moreover, we propose a correction equation as shown in (13). Note that, though a sub-partition has converged and been bypassed, a corrector $B_i^*$ should be updated as the Schur complement of this sub-partition to maintain the accuracy.

$$\boldsymbol{B}_i^* = (\boldsymbol{TOP}_{B_i} - \boldsymbol{D}\boldsymbol{B}_i^{-1}\boldsymbol{C}) \cdot \Delta \boldsymbol{X}_{TOP\_B_i\_new}. \tag{13}$$

Our proof is as follows. Equation (14) shows the sub-matrix for sub-partition $B_i$, where $\Delta \boldsymbol{X}_{B_i}$ is the solution of internal node, $\Delta \boldsymbol{X}_{TOP\_B_i}$ is the solution of top matrix that related to $B_i$.

$$\begin{bmatrix} \boldsymbol{B}_i & \boldsymbol{C} \\ \boldsymbol{D} & \boldsymbol{TOP}_{\_B_i} \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{X}_{B_i} \\ \Delta \boldsymbol{X}_{TOP\_B_i} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{RHS}_{B_i} \\ -\boldsymbol{RHS}_{TOP\_B_i} \end{bmatrix}. \tag{14}$$

If do not bypass this sub-partition, the RHS of top matrix that corresponding to $B_i$ should be

$$\boldsymbol{RHS}_{TOP\_B_i} = \boldsymbol{RHS}_{TOP\_B_i\_old} + \Delta \boldsymbol{RHS}_{TOP\_B_i}, \tag{15}$$

where

$$\Delta \boldsymbol{RHS}_{TOP\_B_i} = \boldsymbol{D} \cdot \Delta \boldsymbol{X}_{B_i\_new} + \boldsymbol{TOP}_{\_B_i} \cdot \Delta \boldsymbol{X}_{TOP\_B_i\_new}. \tag{16}$$

Since condition 2 is satisfied, we have:

$$\boldsymbol{B}_i \Delta \boldsymbol{X}_{B_i\_new} + \boldsymbol{C} \Delta \boldsymbol{X}_{TOP\_B_i\_new} \approx 0. \tag{17}$$

Then we can obtain:

$$\begin{aligned} &\boldsymbol{D} \cdot \Delta \boldsymbol{X}_{B_i\_new} + \boldsymbol{TOP}_{\_B_i} \cdot \Delta \boldsymbol{X}_{TOP\_B_i\_new} \\ &= \boldsymbol{D}(-\boldsymbol{B}_i^{-1}\boldsymbol{C} \cdot \Delta \boldsymbol{X}_{TOP\_B_i\_new}) + \boldsymbol{TOP}_{\_B_i} \cdot \Delta \boldsymbol{X}_{TOP\_B_i\_new} \\ &= (\boldsymbol{TOP}_{\_B_i} - \boldsymbol{D}\boldsymbol{B}_i^{-1}\boldsymbol{C}) \cdot \Delta \boldsymbol{X}_{TOP\_B_i\_new}. \end{aligned} \tag{18}$$

Therefore, the value of the corrector $B_i^*$ is the same as the Schur complement without bypassing as shown in (15). From this proof, it can be found that the accuracy is maintained by the proposed approach.

# 4 NUMERICAL EXAMPLES

In this section, several industrial large-scale circuits from real-world are tested to demonstrate the convergence and efficiency of our proposed method. The proposed method is implemented in a SPICE

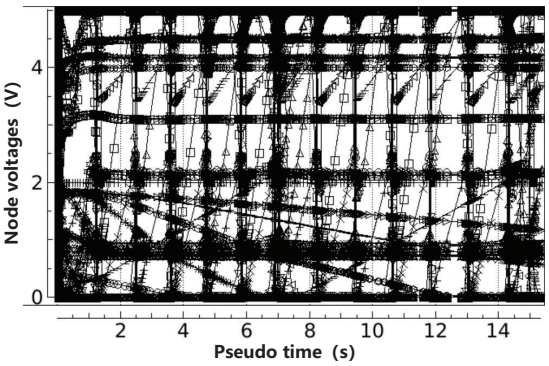


**Figure 3: Solution curves solved by the PTA method. It fails to converge due to oscillation.**

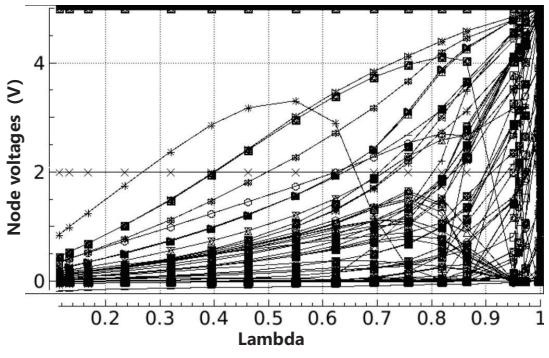


**Figure 4: Solution curves solved by the PALBBD. It reaches the hyperplane $\lambda = 1$ and finally converged.**

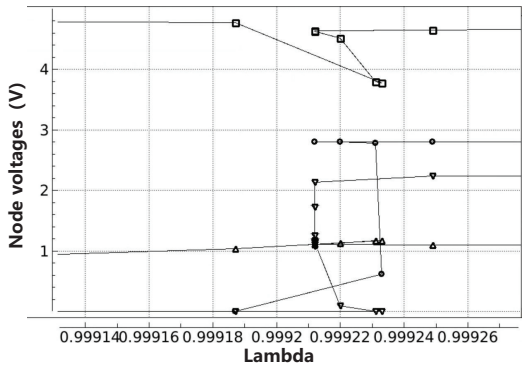


**Figure 5: Solution curves are folded with PALBBD, which cannot be solved by NR, Gmin or source-stepping methods.**

simulator using C++ with OpenMP. Firstly, the convergence performance of the PALBBD is verified and compared with other continuation methods. Secondly, the efficiency of PALBBD is compared with the conventional serial arclength method. After that, the scalability is confirmed through experiments on 2, 4, 8 and 16 threads. Thirdly, the bypassing ratio is verified. All tests are executed on a Linux workstation which has four 2.6GHz Intel Xeon E5-4627 v4 CPUs and 512GB main memory.

## 4.1 Convergence Performance

A large-scale circuit with 1901 devices including 523 MOSFET transistors and 1315 diodes is tested to verify the convergence performance of the PALBBD approach. Due to the heavy positive feedback and strong nonlinearity, the solution curve is folded up and differentially discontinuous. These properties lead to convergence failure in the NR method, source stepping and Gmin stepping method. Moreover, almost all commercial circuit simulators fail to converge in the DC analysis for this circuit. Though the PTA method has been proven with a strong ability to deal with the discontinuous problem, for this circuit, it fails to converge due to oscillation as shown in Fig. 3. The PALBBD converges after 116 arclength iterations. The solution curve of PALBBD is shown in Fig. 4 and the $\lambda$ finally reaches 1. The main reason is that the arclength traces the solution curve continuously and makes it possible for $\lambda$ to fall back as shown in Fig. 5.

## 4.2 Parallel Speedup and Scalability

Comparisons are illustrated between a conventional serial arclength method without BBD-form partition and the PALBBD with 1, 2, 4, 8 and 16 threads. Ten real-world circuits are tested, and the speedup ratios are shown in Table 1. The total number of elements vary from 12,586 to 3,159,178 and the average speedup ratio of serial PALBBD over conventional serial arclength method is 1.95X (the maximum speedup ratio is 3.72X). What's more, the average speedup ratio of PALBBD on 16 (8, 4, 2) threads over serial version is 7.72X (5.52X, 3.36X, 1.63X) showing strong scalability, while the maximum speedup ratio reaches 12.56X (6.73X, 3.94X, 1.78X). Moreover, another 62 large-scale circuits are also tested and results are shown in Fig. 6, the average (maximum) speedup ratio of PALBBD on 16 threads over conventional serial arclength is 10.2X (73.93X).

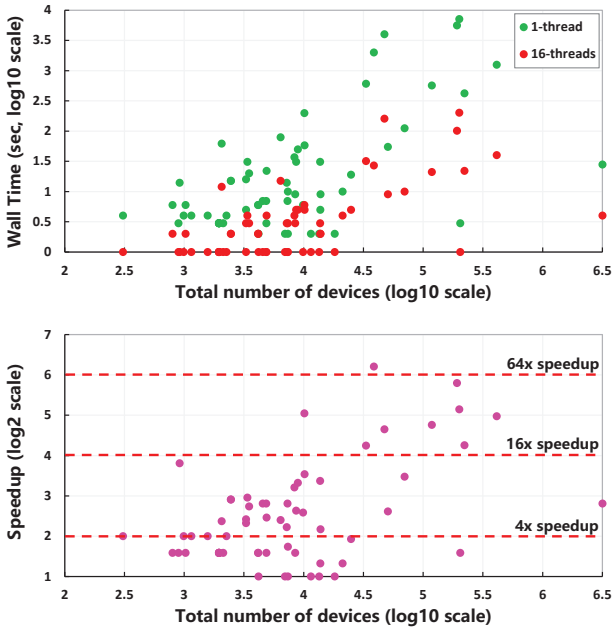


**Figure 6: Efficiency comparison of PALBBD on 16 threads and conventional serial method for 62 different circuits.**

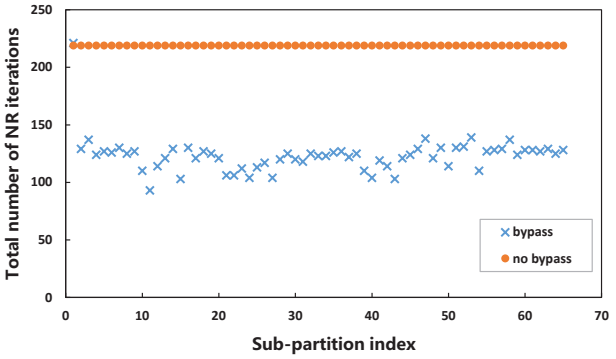


**Figure 7: Comparisons of the proposed bypassing and the conventional scheme on the total NR iteration numbers for each sub-partition using a 512Kx40bit SRAM circuit.**

## 4.3 Efficiency of Bypassing

A 512Kx40bit SRAM is shown here to demonstrate the accelerating effect of the proposed bypassing method. This circuit has 1151188 resistors, 10 diodes and 135,129 MOSFETs. The circuit is partitioned into 64 sub-partitions. After 84 arclength iterations, the circuit converged to the solution. Fig. 7 shows the comparisons of total number of NR iteration steps for each sub-partition between our proposed bypassing scheme and the one without bypassing. As shown in Fig. 8, many sub-partitions only need 1 or 2 NR iterations at each arclength step. The top-partition and a few difficult sub-partitions need 3 or 4 NR iterations. The speedup of the proposed bypassing strategy relies on the type of circuit and the partitioning method. As we tested, the memory, PLL, and some mixed analog/digital circuits can achieve an average acceleration of 35% and a maximum acceleration of 45%.

**Table 1: Performance of parallel DC analysis for industrial circuits on the 2.6 GHz Intel(R) Xeon(R) CPU**

| Circuit | Scale[1] | Nonlinear devices[2] | Blocks[3] | Serial arclength (s)[4] | Serial PALBBD (s)[5] | Speedup ratio[6] | Scalability[7] | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | p2 | p4 | p8 | p16 |
| sram_512K | 3159178 | 3159136 | 16 | 29 | 21.22 | 1.37 | 1.48 | 3.81 | 5.81 | 7.50 |
| alter_post_RC | 70474 | 10521 | 16 | 99 | 97.13 | 1.02 | 1.66 | 3.55 | 6.36 | 6.97 |
| sram16k | 114675 | 114646 | 64 | 254 | 121.25 | 2.09 | 1.45 | 2.72 | 4.90 | 8.19 |
| PADC | 12586 | 11372 | 16 | 369 | 194.66 | 1.90 | 1.60 | 3.94 | 6.48 | 10.51 |
| Divider_post_RC | 53101 | 3504 | 8 | 41 | 16.95 | 2.42 | 1.59 | 3.17 | 3.80 | 3.91 |
| kp_mip | 36277 | 34623 | 16 | 205 | 92.66 | 2.21 | 1.78 | 3.31 | 6.02 | 7.19 |
| kp_yt16_case2 | 36405 | 34623 | 16 | 215 | 84.72 | 2.54 | 1.74 | 3.24 | 5.76 | 7.94 |
| access_cmg_post_C | 48326 | 24405 | 32 | 918 | 781.08 | 1.18 | 1.63 | 3.64 | 6.73 | 12.56 |
| normalx5 | 20011 | 20010 | 128 | 255 | 68.50 | 3.72 | 1.68 | 3.29 | 5.96 | 8.83 |
| testx3_post | 78206 | 4303 | 32 | 54 | 52.13 | 1.04 | 1.73 | 2.95 | 3.37 | 3.64 |
| **Average speedup ratio** | | | | | | | 1.63 | 3.36 | 5.52 | 7.72 |

[1]The number of total elements (without capacitor and inductor).
[2]The number of nonlinear devices.
[3]The number of sub-partitions of every circuit.
[4]The wall time of conventional arclength method on single thread.
[5]The wall time of PALBBD on single thread.
[6]Speedup of serial PALBBD over conventional method.
[7]Speedup of parallel PALBBD using 2,4,8,16 threads over serial version.

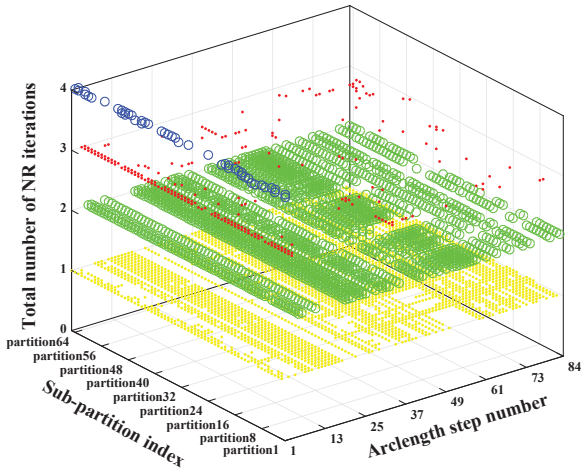


**Figure 8: The number of NR iterations for each sub-partition at each arclength step by using the proposed bypassing strategy. The four colors represent those sub-partitions converged after 1, 2, 3 or 4 NR iterations respectively at current arclength step.**

## 5 CONCLUSIONS

We presented the PALBBD approach exploiting the BBD-form matrix with an accurate bypassing technique. To our knowledge, this is the first work on the parallel arclength continuation method for fast and accurate DC analysis in circuit simulation. The performance is verified by industrial large-scale post-layout circuits. Compared with the serial method, PALBBD achieves up to 73.93 times of acceleration on 16 threads. The proposed approach could provide the possibility of an adaptive DC approach in the future, where each sub-partition has different parameters and algorithms.

## ACKNOWLEDGEMENT

We deeply appreciate the invaluable comments from the reviewers. Weifeng Liu is the corresponding author of this paper. This work was supported by the National Natural Science Foundation of China under Grant No. 61972415 and Science Foundation of China University of Petroleum, Beijing under Grant No. 2462020YXZZ024.

## REFERENCES

[1] D. E. C. Udave, J. Ogrodzki and G. M. A. de Anda, "DC large-Scale Simulation of Nonlinear Circuits on Parallel Processors," IJET, vol. 58(3), pp. 285–295, 2012.
[2] X. Wu, Z. Jin, D. Niu and Y. Inoue, "PTA method using numerical integration algorithms with artificial damping for solving nonlinear DC circuits," IEICE NOLTA, vol.E5-N, No.4, pp. 512–522, Oct. 2014.
[3] K. Yamamura, T. Sekiguchi and Y. Inoue, "A fixed-point homotopy method for solving modified nodal equations," in IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, vol. 46, no. 6, pp. 654–665, June 1999.
[4] D. Niu, Y. Inoue, Z. Jin and X. Wu, "A netlist implementation of the Newton fixed-point homotopy method for MOS transistor circuits," IEEE MWSCAS, Fort Collins, CO, pp. 1–4, 2015.
[5] K. W. Chan, "Parallel algorithms for direct solution of large sparse power system matrix equations," in IEE Proceedings - Generation, Transmission and Distribution, vol. 148, no. 6, pp. 615–622, Nov. 2001.
[6] D. P. Koester, S. Ranka and G. C. Fox, "Parallel block-diagonal-bordered sparse linear solvers for electrical power system applications," Proceedings of Scalable Parallel Libraries Conference, Mississippi State, MS, USA, pp. 195–203, 1993.
[7] X. Chen, Y. Wang and H. Yang, "Parallel sparse direct solver for integrated circuit simulation," Springer International Publishing, 2017.
[8] C. Aykanat, A. Pinar and U.V. Catalyürek, "Permuting sparse rectangular matrices into Block Diagonal form," SIAM J.Sci. Comput., vol. 25(6), pp. 1860–1879, 2004.
[9] I. S. Duff and J. A. Scott, "Stabilized bordered block diagonal forms for parallel sparse solvers," Parallel Comput., vol. 31(3-4), pp. 275–289, 2005.
[10] E .Ikeno and A. Ushida, "The arc-length method for the computation of characteristic curves," IEEE TCAS, vol. 23(3), pp. 181–183, 1976.
[11] K. Yamamura and K. Adachi, "A modified predictor-corrector method for tracing solution curves," IEEE APCCAS, Jeju, pp. 372–375, 2016.
[12] N. Frohlich, B. M. Riess, U. A. Wever and Q. Zheng, "A new approach for parallel simulation of VLSI circuits on a transistor level," IEEE TCS. I: Fundam. Theory Appl., vol. 45(6), pp. 601—613, 1998.
[13] L. Trajkovic, R. C. Melville and S. -. Fang, "Improving DC convergence in a circuit simulator using a homotopy method," IEEE 1991 Custom Integrated Circuits Conference, pp. 1–4, 1991.
[14] Benk, J. and Denk, G. and Waldherr, K. "A holistic fast and parallel approach for accurate transient simulations of analog circuits," Journal of Mathematics in Industry, vol. 7, No. 12, 2017.
[15] J. Zhao , Y. Wen , Y. Luo , Z. Jin , W. Liu and Z. Zhou, "SFLU: Synchronization-free sparse LU factorization for fast circuit simulation on GPUs," DAC, San Francisco, 2021.
[16] L. Trajkovic, "DC operating points of transistor circuits," Nonlinear Theory and Its Applications IEICE, pp. 287–300, 2012.
[17] K. Yamamura and T. Shimada, "An efficient variable-gain homotopy method for finding DC operating points of transistor circuits," IEEE Asia Pacific Conference on Circuits and Systems, Chengdu, China, October 26-30, pp. 235–238, 2018.
[18] K. Yamamura and W. Kuroki, "An efficient and globally convergent homotopy method for finding DC operating points of nonlinear circuits," In Proceedings of the Conference on Asia South Pacific Design Automation (ASP-DAC), Yokohama, Japan, January 24-27, pp. 408–41, 2006.