More Details for VPP-ART

Rubing Huang, *Senior Member, IEEE*, Chenhui Cui, Dave Towey, *Senior Member, IEEE*,
Weifeng Sun, Junlong Lian

1 VPP-ART ALGORITHMS

In this section, we provide the detailed descriptions of VPP-ART algorithms, and then discuss the nearest neighbor threshold for VPP-ART.

1.1 Description

Algorithm 1 presents the pseudo-code of VPP-ART; Algorithm 2 describes the pseudo-code of **InsertTCIntoVPtree**$(tc, \lambda, \varepsilon, node)$, i.e., the Insertion/Promotion function; while Algorithm 3 gives the pseudo-code of **GetMinDistFromVPtree**$(c_j, node)$, i.e., the search process to get the minimal distance.

- *R. Huang is with the School of Computer Science and Engineering, Macau University of Science and Technology, Taipa, Macau 999078, China. E-mail: rbhuang@must.edu.mo.*
- *C. Cui and J. Lian are with the School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang, Jiangsu 212013, China. E-mail: {2211908012, 2211908018}@stmail.ujs.edu.cn.*
- *D. Towey is with the School of Computer Science, University of Nottingham Ningbo China, Ningbo, Zhejiang 315100, China. E-mail: dave.towey@nottingham.edu.cn.*
- *W. Sun is with the School of Big Data and Software Engineering, Chongqing University, Chongqing, 401331, China. E-mail: weifeng.sun@cqu.edu.cn.*

Algorithm 1: VPP-ART

Inputs:
The candidate set size, k;
The number of dimensions in the SUT input domain, d;
The capacity of a leaf node, λ;
The partitioning parameter, ε;
Output:
The VP-tree with executed test cases, T;

Set $T \leftarrow \{\}$; /* *Initially, VP-tree for storing the executed test cases is empty.* */
Set $C \leftarrow \{\}$; /* *Initially, candidate test case set C, for storing k candidates in each round, is empty.* */
Set $min_{dist}[k] \leftarrow \infty$; /* *Initially, the minimum distance between each candidate test case and T is set to infinity.* */
Randomly generate a test case tc from the input domain
Execute tc;
while (*No termination condition is satisfied*)
  InsertTCIntoVPtree$(tc, \lambda, \varepsilon, node)$;
  Randomly generate k candidate test cases $c_1, c_2, \cdots, c_k$ from the input domain, then set $C \leftarrow \{c_1, c_2, \cdots, c_k\}$;
  for (each candidate $c_j \in C$, where $j = 1, 2, \cdots, k$)
    Set $min_{dist}[j] \leftarrow$ **GetMinDistFromVPtree**$(c_j, node)$;
  end_for
  Select c_{best} from $c_1, c_2, \cdots, c_k$ such that it has the maximum distance to its nearest executed test case;
  Set $tc \leftarrow c_{best}$ and execute tc;
end_while
return T;

Algorithm 2: InsertTCIntoVPtree$(tc, \lambda, \varepsilon, node)$

Inputs:
The executed test case need to insert into the tree, tc;
The capacity of a leaf node, λ;
The number of subsets partition, ε;
A node of the VP-tree, $node$;
Output:
Success flag, FALSE or TRUE;

if ($node$ *is* tc-QBN (a leaf node))
  if ($|tc\text{-QBN}| < \lambda$)
    Insert tc into tc-QBN;
    Set tc-BN $\leftarrow$ tc-QBN;
    return TRUE;
  else if ($|tc\text{-QBN}| = \lambda$) /* ***Node promotion strategy*** */
    Randomly select a test case from tc-QBN, as the vantage test case, vp;
    for (each test case $p \in tc$-QBN $\cup\{tc\}$)
      Calculate $dist(p, vp)$;
      Sort the test cases in ascending order according to the distance values;
    end_for
    Calculate the boundary distance values μ_i according to Eq. (2);
    Calculate σ of current node, using Eq. (6);
    Partition the input domain into ε sub-domains;
    /* *Each sub-domain contains approximately the same number of test cases* */
    Reorganize all the test cases to the children nodes of this new common node;
    if (*allocate the new* tc-BN)
      return TRUE;
    end_if
  end_if
else
  Calculate $dist(vp, tc)$;
  if ($\mu_i < dist(vp, tc) \leq \mu_{i+1}$ where $0 \leq i \leq \varepsilon - 2$)
    InsertTCIntoVPtree$(tc, \lambda, \varepsilon, node.i$-th child node);
  end_if
  if ($dist(vp, tc) > \mu_{\varepsilon-1}$)
    InsertTCIntoVPtree$(tc, node.(\varepsilon - 1)$-th child node);
  end_if
end_if

1.2 Nearest Neighbor Threshold for VPP-ART

The threshold value σ is the key to the search algorithm. It can be used to reduce the NN search effort such that the NNs of a query point are always very close to the query point itself — in other words, the smallest possible value

Algorithm 3: GetMinDistFromVPtree$(c_j, node)$

Inputs:
A candidate test case, c_j;
A node of the VP-tree, $node$;
Output:
The minimum distance between c_j and E, $min_{dist}[j]$;

$min_{dist}[j] \leftarrow \infty$;
if ($node$ *is leaf node*)
  for (each test case p in $node$)
    if ($dist(c_j, p) < min_{dist}[j]$)
      $min_{dist}[j] \leftarrow dist(c_j, p)$;
    end_if
  end_for
  return $min_{dist}[j]$;
else
  Calculate $dist(c_j, vp)$;
  for ($0 \leq i \leq \varepsilon - 2$))
    if ($\mu_i - \sigma < dist(vp, c_j) \leq \mu_{i+1} + \sigma$)
      Get σ of current common node;
      GetMinDistFromVPtree($c_j, node.i$-th child node);
    end_if
  end_for
  if ($dist(c_j, vp) > \mu_{\varepsilon-1}$)
    GetMinDistFromVPtree($c_j, node.(\varepsilon - 1)$-th child node);
  end_if
end_if

of σ can be used. Chiueh *et al.* [29] proposed a minimum possible value of σ, which we adopted and modified to match our partitioning strategy. Upper ($u[i]$) and lower ($l[i]$) bounds exist for distances in the i-th subset $\mathcal{D}_i$ of the partitioned input domain: $u[i]$ is the distance from the farthest test case in the current subset to its corresponding vantage test case, and $l[i]$ is the distance from the nearest test case. When traversing the VP-tree, it is not necessary to explore the nearest executed test cases of a candidate c in $\mathcal{D}_i$ if $dist(c, vp) > u[i] + \sigma$, or $dist(c, vp) < l[i] - \sigma$. This use of σ guarantees that $dist(c, vp)$ will fall within at least one of the ranges in $\left[l[i] - \sigma, u[i] + \sigma\right]$: This means that the search operation can be performed (and completed) in one sub-domain of the entire input domain. For each common node, the upper and lower bounds of ε partitioned subsets are computed as: $\sigma_i = (l[i] - u[i-1])/2$, and the default value of σ (for that node) is chosen to be the maximum value from $(\varepsilon - 1)$ values, as determined by:

$$\sigma = \max_{i=1}^{\varepsilon-1} \sigma_i. \tag{1}$$

Based on the relationship among σ, μ, and $dist(c, vp)$, VPP-ART can start from the root node of the modified VP-tree and recursively search downwards until it finds the leaf node that contains the nearest executed test case of the current candidate.

Combined with the partitioning boundary values, the last sub-domain ($\mathcal{D}_{\varepsilon-1}$) is defined as:

$$\mu_i - \sigma < d(q, vp) \leq \mu_{i+1} + \sigma, \tag{2}$$

which is not covered by the search process. The reason for this is that the possible values of i can only be from 0 to $\varepsilon - 2$. When $dist(q, vp) > \mu_{\varepsilon-1}$, the NN search process cannot be executed in $\mathcal{D}_{\varepsilon-1}$. To solve this, $\mathcal{D}_{\varepsilon-1}$ is treated separately: If $dist(q, vp) > \mu_{\varepsilon-1}$, then an exhaustive search in $\mathcal{D}_{\varepsilon-1}$ is conducted to obtain the query point q, and to get the NN distance in $\mathcal{D}_{\varepsilon-1}$. This means that the NN of q — which is located at the boundary of $\mathcal{D}_{\varepsilon-2}$ and $\mathcal{D}_{\varepsilon-1}$ — may be in $\mathcal{D}_{\varepsilon-2}$, while VPP-ART only searches in $\mathcal{D}_{\varepsilon-1}$, and finds an approximate NN, rather than an exact NN.

2 VPP-ART Parameter Settings

The VPP-ART performance is strongly impacted by two parameters: the partitioning parameter, ε; and the maximum test case capacity of a leaf node, λ. These two values play important roles in the partitioning of the input domain, and also limit the amount of executed test cases in each sub-domain, which can affect the accuracy of the NN returned by VPP-ART. When VPP-ART performs the NN search in one sub-domain, the approximate NN is identified. When there are more executed test cases in this sub-domain — which is directly influenced by the static values of ε and λ — then VPP-ART will have a greater probability of finding a more accurate NN (of course, it may also be an approximate NN due to the construction of the modified VP-tree). If the approximate NN returned by VPP-ART is similar to the exact NN, the failure-detection effectiveness of VPP-ART and FSCS-ART will be comparable. Therefore, this section focuses on the influence of different $\langle \varepsilon, \lambda \rangle$ parameter pair values on VPP-ART. The specific parameter settings for the simulations were as follows:

- Dimension: $d = 1, 2, 3, 4, 5, 8, 10$;
- Failure rate: $\theta = 0.0005$;
- Partitioning parameter, $\varepsilon = 2, 3, 4, 5$;
- Maximum test case capacity of a leaf node, $\lambda = 10, 15, 20, 25, 30, 35, 40, 45, 50$.

Here, we briefly explain the reason for setting the failure rate (θ) to 0.0005. The failure rate reflects the proportion of inputs that can cause software failures. Theoretically, the number of test cases used by RT to find the first failure is the reciprocal of θ. When $\theta = 0.0005$, RT should need (on average) 2000 test cases to find a failure. This value is appropriate for this experiment. Specifically, when the number of test cases is very small, the breadth or depth of the constructed VP tree does not reflect the advantages of the algorithm. In other words, the cost of constructing the VP tree to partition the space may be greater than the cost of finding the NN. In addition, when there are too many test cases, the verification time can be huge. These conditions are not conducive to the exploration of parameters. Therefore, we identified $\theta = 0.0005$ as an appropriate failure rate.

Table 1 presents the *ART F-ratio* simulation results of VPP-ART for the different parameter values. Based on the data in the table, some observations can be summarized as follows:

(1) As the maximum test case capacity of leaf nodes (λ) increases, the VPP-ART *ART F-ratio* differences (when $1 \leq d \leq 5$) are not significant, but are, on the whole, slightly higher than FSCS-ART; For $d = 8, 10$, the *ART F-ratio* values increase gradually, and are lower than FSCS-ART when λ is small. This shows that changes in λ have little impact on the failure-detection effectiveness of VPP-ART in low dimensional input domains, but can have a great impact in high dimensions. Therefore, a small λ value can effectively ensure VPP-ART failure-detection effectiveness

TABLE 1
ART F-ratio results of VPP-ART with different $\langle\varepsilon, \lambda\rangle$ parameter pair values

Partitioning Number (ε)	Dimension (d)	ART F-ratio									
		FSCS-ART	*VPP-ART*								
			$\lambda = 10$	$\lambda = 15$	$\lambda = 20$	$\lambda = 25$	$\lambda = 30$	$\lambda = 35$	$\lambda = 40$	$\lambda = 45$	$\lambda = 50$
$\varepsilon = 2$	$d = 1$	0.5564	0.5706	0.5652	0.5523	0.5611	0.5675	0.5747	0.5631	0.5634	0.5702
	$d = 2$	0.6391	0.6790	0.6584	0.6752	0.6712	0.6645	0.6547	0.6641	0.6516	0.6440
	$d = 3$	0.7549	0.8056	0.7890	0.7809	0.7787	0.7964	0.8053	0.7957	0.8096	0.7970
	$d = 4$	0.9033	0.9611	0.9329	0.9243	0.9391	0.9506	0.9539	0.9280	0.9368	0.9462
	$d = 5$	1.0462	1.0824	1.0945	1.0984	1.0852	1.0978	1.1147	1.1260	1.1200	1.1069
	$d = 8$	1.8607	1.7050	1.6886	1.7668	1.7673	1.8200	1.7956	1.7711	1.8460	1.8289
	$d = 10$	2.6138	2.4504	2.4896	2.6366	2.6495	2.6355	2.6268	2.7046	2.7357	2.7719
$\varepsilon = 3$	$d = 1$	0.5564	0.5555	0.5605	0.5686	0.5584	0.5557	0.5573	0.5599	0.5514	0.5582
	$d = 2$	0.6391	0.6510	0.6601	0.6447	0.6559	0.6377	0.6661	0.6459	0.6725	0.6523
	$d = 3$	0.7549	0.7553	0.7534	0.8083	0.7832	0.7753	0.7663	0.8063	0.8037	0.7940
	$d = 4$	0.9033	0.9390	0.9448	0.9510	0.9385	0.9385	0.9345	0.9355	0.9484	0.9421
	$d = 5$	1.0462	1.0605	1.0890	1.0709	1.0698	1.1085	1.0876	1.0997	1.1178	1.1053
	$d = 8$	1.8607	1.6631	1.8007	1.7354	1.7543	1.8558	1.8406	1.8833	1.8870	1.8493
	$d = 10$	2.6138	2.2916	2.4821	2.4765	2.5670	2.6265	2.6319	2.7906	2.7058	2.7080
$\varepsilon = 4$	$d = 1$	0.5564	0.5577	0.5636	0.5621	0.5613	0.5529	0.5726	0.5567	0.5607	0.5588
	$d = 2$	0.6391	0.6570	0.6587	0.6590	0.6739	0.6534	0.6296	0.6340	0.6448	0.6529
	$d = 3$	0.7549	0.7617	0.7680	0.7571	0.7604	0.7659	0.7519	0.7900	0.7709	0.7754
	$d = 4$	0.9033	0.9144	0.9159	0.9272	0.9086	0.9418	0.9572	0.9427	0.9418	0.9393
	$d = 5$	1.0462	1.0896	1.0891	1.0734	1.0727	1.1486	1.0854	1.0964	1.0948	1.1133
	$d = 8$	1.8607	1.6900	1.6744	1.8339	1.8043	1.8048	1.8067	1.8578	1.9174	1.9717
	$d = 10$	2.6138	2.3144	2.3562	2.6492	2.6325	2.8174	2.6989	2.8207	2.8412	2.9358
$\varepsilon = 5$	$d = 1$	0.5564	0.5573	0.5542	0.5507	0.5822	0.5566	0.5639	0.5558	0.5585	0.5496
	$d = 2$	0.6391	0.6467	0.6594	0.6416	0.6360	0.6536	0.6427	0.6412	0.6433	0.6447
	$d = 3$	0.7549	0.7912	0.7747	0.7547	0.7745	0.7674	0.7518	0.7610	0.7671	0.7726
	$d = 4$	0.9033	0.9276	0.9167	0.9024	0.9194	0.9582	0.9239	0.9559	0.9592	0.9391
	$d = 5$	1.0462	1.0932	1.0849	1.0923	1.1254	1.1063	1.1072	1.1258	1.1370	1.1280
	$d = 8$	1.8607	1.7753	1.7982	1.8214	1.8652	1.8581	2.0066	1.9264	1.9821	1.9547
	$d = 10$	2.6138	2.4743	2.5811	2.6273	2.7442	2.7095	2.8409	2.9523	2.8931	2.9514

in low dimensions, and improve FSCS-ART performance in high dimensions.

(2) When the partition parameter (ε) increases above 2, the VPP-ART *ART F-ratio* values are not significantly different when $1 \leq d \leq 5$. When $d = 8, 10$, the *ART F-ratio* values show a trend of first decreasing, and then increasing; an inflection point appears when $\varepsilon = 3$. Similar to λ, changes in ε have little impact on VPP-ART failure-detection effectiveness in low dimensions, but show a certain change trend in the high dimensions. Therefore, a smaller ε value can enhance the VPP-ART performance.

Based on the above, the parameter pair $\langle\varepsilon, \lambda\rangle$ were assigned $\langle 3, 10\rangle$ for the simulations and experiments.

3 Detailed Experimental Data

In the tables in this section, the ***blue bold*** typeface indicates the minimum value of *ART F-ratio*, *F-measure* or *F-time* across the several ART algorithms; and the ***red bold*** means that the *p-value* of the comparison between VPP-ART and the corresponding ART algorithm is less than 0.05, indicating significance.

3.1 Comparisons of Failure-Detection Effectiveness

3.1.1 Answer to RQ1 - Part 1: Results of Simulations

Tables 2 to 4 provide the detailed ART F-ratio simulation results and the statistical comparisons of VPP-ART against other ART algorithms, according to the block, strip, and point patterns, respectively.

TABLE 2
ART F-ratio results and statistical analysis comparisons among VPP-ART and other ART algorithms for *block patterns*

Dimension (d)	Failure Rate (θ)	ART F-ratio					Statistical Analysis							
							vs. FSCS-ART		*vs. Naive-KDFC*		*vs. SemiBal-KDFC*		*vs. LimBal-KDFC*	
		VPP-ART	*FSCS-ART*	*Naive-KDFC*	*SemiBal-KDFC*	*LimBal-KDFC*	*p-value*	*effect size*	*p-value*	*effect size*	*p-value*	*effect size*	*p-value*	*effect size*
	0.01	**0.5634**	0.5729	0.5664	0.5714	0.5658	0.3679	0.5067	0.7899	0.5020	0.4101	0.5061	0.5294	0.5047
	0.005	0.5666	0.5633	0.5670	0.5696	**0.5619**	0.8994	0.4991	0.7826	0.5021	0.4768	0.5053	0.8362	0.4985
	0.002	0.5639	0.5683	0.5665	0.5723	**0.5605**	0.5704	0.5042	0.9981	0.5000	0.5034	0.5050	0.7817	0.4979
$d=1$	0.001	0.5634	0.5720	**0.5549**	0.5690	0.5617	0.6041	0.5039	0.1890	0.4902	0.5659	0.5043	0.7915	0.4980
	0.0005	**0.5555**	0.5564	0.5629	0.5556	0.5619	0.5919	0.5040	0.4510	0.5056	0.7454	0.5024	0.4847	0.5052
	0.0002	**0.5520**	0.5700	0.5658	0.5662	0.5614	0.0602	0.5140	0.1353	0.5111	0.2506	0.5086	0.2506	0.5086
	0.0001	0.5766	0.5765	**0.5527**	0.5545	0.5569	0.8939	0.4990	**0.0327**	0.4841	0.0559	0.4857	0.1115	0.4881
	0.01	**0.6820**	0.6911	0.6822	0.6904	0.6953	0.1867	0.5098	0.4907	0.5051	0.1303	0.5113	0.0805	0.5130
	0.005	0.6750	0.6613	**0.6561**	0.6635	0.6671	0.7798	0.4979	0.3619	0.4932	0.6590	0.4967	0.8597	0.4987
	0.002	0.6712	**0.6536**	0.6574	0.6633	0.6561	0.5933	0.4960	0.7498	0.4976	0.5793	0.5041	0.5865	0.4959
$d=2$	0.001	0.6742	0.6573	**0.6449**	0.6557	0.6595	0.5260	0.4953	0.1795	0.4900	0.6460	0.4966	0.6193	0.4963
	0.0005	0.6510	**0.6391**	0.6525	0.6484	0.6492	0.9636	0.4997	0.2177	0.5092	0.4341	0.5058	0.3357	0.5072
	0.0002	0.6489	**0.6268**	0.6409	0.6414	0.6388	0.2310	0.4911	0.9428	0.4995	0.8521	0.4986	0.5000	0.4950
	0.0001	**0.6244**	0.6248	0.6531	0.6389	0.6313	0.5895	0.5040	**0.0030**	0.5222	0.0847	0.5129	0.4206	0.5060
	0.01	0.8840	0.8641	0.8431	0.8504	**0.8391**	0.6879	0.5030	0.2535	0.4915	0.7570	0.4977	0.2964	0.4922
	0.005	0.8391	0.8314	**0.8176**	0.8195	0.8177	0.3398	0.5071	0.9302	0.5007	0.6204	0.5037	0.5875	0.4960
	0.002	0.8214	0.7847	**0.7778**	0.7948	0.8052	0.2653	0.4917	0.1122	0.4882	0.5126	0.4951	0.3310	0.4928
$d=3$	0.001	0.8189	0.7735	**0.7720**	0.7735	0.7772	0.1464	0.4892	0.0981	0.4877	0.1562	0.4894	0.0703	0.4865
	0.0005	0.7553	0.7549	**0.7504**	0.7618	0.7615	0.2115	0.5093	0.3457	0.5070	0.2778	0.5081	0.2614	0.5084
	0.0002	0.7881	0.7499	0.7603	**0.7441**	0.7464	0.5469	0.4955	0.5871	0.4960	0.1881	0.4902	0.2464	0.4914
	0.0001	0.7688	0.7358	**0.7252**	0.7518	0.7387	0.6934	0.4971	0.1312	0.4887	0.7021	0.4971	0.2108	0.4907
	0.01	1.0886	1.0786	1.0739	1.0711	**1.0666**	0.9147	0.5008	0.6676	0.5032	0.9517	0.4995	0.9651	0.5003
	0.005	1.0523	1.0272	1.0350	**1.0200**	1.0202	0.5394	0.4954	0.8604	0.5013	0.4641	0.4945	0.5352	0.4954
	0.002	0.9948	0.9606	**0.9497**	0.9711	0.9754	0.8243	0.4983	0.5681	0.4957	0.9008	0.5009	0.9820	0.4998
$d=4$	0.001	0.9398	0.9155	**0.9122**	0.9190	0.9366	0.7930	0.5020	0.4660	0.4946	0.8987	0.4991	0.8987	0.5009
	0.0005	0.9390	0.9033	0.8908	**0.8904**	0.9067	0.1379	0.4889	0.1067	0.4880	**0.0464**	0.4852	0.2141	0.4907
	0.0002	0.8965	0.8522	**0.8494**	0.8651	0.8708	0.0963	0.4876	0.2810	0.4920	0.1513	0.4893	0.6204	0.4963
	0.0001	0.8857	0.8357	**0.8234**	0.8687	0.8491	0.3925	0.4936	0.3509	0.4930	0.4141	0.5061	0.8148	0.5017
	0.01	1.3417	1.3346	1.3416	1.3268	**1.3209**	0.6070	0.5038	0.9107	0.5008	0.8170	0.4983	0.5754	0.5042
	0.005	1.2809	1.2694	1.2638	1.2632	**1.2550**	0.7747	0.4979	0.8919	0.4990	0.6168	0.4963	0.7876	0.5020
	0.002	1.1671	1.1661	1.1932	1.1685	**1.1550**	0.3047	0.5077	0.1059	0.5121	0.5585	0.5044	0.5848	0.5041
$d=5$	0.001	1.1130	1.1097	1.1185	1.1317	**1.0850**	0.9833	0.5002	0.6000	0.5039	0.2411	0.5087	0.6566	0.4967
	0.0005	1.0605	1.0462	1.0498	1.0584	**1.0217**	0.7815	0.5021	0.7631	0.5022	0.8088	0.5018	0.4692	0.4946
	0.0002	1.0404	1.0156	**0.9930**	1.0215	1.0054	0.8649	0.4987	0.5564	0.4956	0.5235	0.5048	0.5712	0.4958
	0.0001	1.0223	0.9935	0.9833	**0.9810**	0.9867	0.8739	0.5012	0.7783	0.4979	0.3574	0.4931	0.4456	0.4943
	0.01	**2.4832**	2.6802	2.6413	2.6390	2.5701	**0.0000**	0.5319	**0.0030**	0.5221	**0.0046**	0.5211	**0.0026**	0.5225
	0.005	**2.2558**	2.4032	2.4134	2.3672	2.2685	**0.0081**	0.5197	**0.0010**	0.5246	**0.0079**	0.5198	0.1401	0.5110
	0.002	**1.9843**	2.1176	2.1177	2.0986	2.0333	0.0672	0.5136	0.0511	0.5145	0.1144	0.5118	0.1073	0.5120
$d=8$	0.001	**1.7889**	1.9526	1.9312	1.9525	1.8306	**0.0004**	0.5263	**0.0027**	0.5224	**0.0068**	0.5202	0.1988	0.5096
	0.0005	**1.6631**	1.8607	1.8474	1.8163	1.7219	**0.0000**	0.5352	**0.0000**	0.5303	**0.0015**	0.5237	0.1321	0.5112
	0.0002	**1.5212**	1.7096	1.7099	1.6956	1.5956	**0.0000**	0.5340	**0.0000**	0.5314	**0.0000**	0.5318	0.1752	0.5101
	0.0001	**1.3741**	1.6325	1.5772	1.6110	1.5027	**0.0000**	0.5510	**0.0000**	0.5452	**0.0000**	0.5502	**0.0003**	0.5267
	0.01	**3.3475**	3.9718	3.9114	3.9454	3.8735	**0.0000**	0.5539	**0.0000**	0.5552	**0.0000**	0.5567	**0.0000**	0.5583
	0.005	**3.1357**	3.5995	3.4993	3.5591	3.4597	**0.0000**	0.5456	**0.0000**	0.5342	**0.0000**	0.5447	**0.0000**	0.5434
	0.002	**2.8812**	3.1565	3.1775	3.1184	2.9093	**0.0000**	0.5372	**0.0000**	0.5367	**0.0000**	0.5306	**0.0239**	0.5168
$d=10$	0.001	**2.6297**	2.8741	2.8712	2.9142	2.6868	**0.0004**	0.5263	**0.0011**	0.5243	**0.0005**	0.5259	0.1794	0.5100
	0.0005	**2.2916**	2.7049	2.7083	2.7002	2.4310	**0.0000**	0.5432	**0.0000**	0.5466	**0.0000**	0.5443	0.0728	0.5134
	0.0002	**1.9882**	2.4118	2.3827	2.4727	2.1770	**0.0000**	0.5536	**0.0000**	0.5528	**0.0000**	0.5614	**0.0007**	0.5252
	0.0001	**1.8902**	2.2568	2.2235	2.3156	2.0289	**0.0000**	0.5481	**0.0000**	0.5466	**0.0000**	0.5549	**0.0150**	0.5181

TABLE 3
ART F-ratio results and statistical analysis comparisons among VPP-ART and other ART algorithms for *strip patterns*

Dimension (d)	Failure Rate (θ)	ART F-ratio					Statistical Analysis							
							vs. FSCS-ART		*vs. Naive-KDFC*		*vs. SemiBal-KDFC*		*vs. LimBal-KDFC*	
		VPP-ART	*FSCS-ART*	*Naive-KDFC*	*SemiBal-KDFC*	*LimBal-KDFC*	*p-value*	*effect size*	*p-value*	*effect size*	*p-value*	*effect size*	*p-value*	*effect size*
$d = 1$	0.01	**0.5634**	0.5729	0.5664	0.5714	0.5658	0.3679	0.5067	0.7899	0.5020	0.4101	0.5061	0.5294	0.5047
	0.005	0.5666	0.5633	0.5670	0.5696	**0.5619**	0.8994	0.4991	0.7826	0.5021	0.4768	0.5053	0.8362	0.4985
	0.002	0.5639	0.5683	0.5665	0.5723	**0.5605**	0.5704	0.5042	0.9981	0.5000	0.5034	0.5050	0.7817	0.4979
	0.001	0.5634	0.5720	**0.5549**	0.5690	0.5617	0.6041	0.5039	0.1890	0.4902	0.5659	0.5043	0.7915	0.4980
	0.0005	**0.5555**	0.5564	0.5629	0.5556	0.5619	0.5919	0.5040	0.4510	0.5056	0.7454	0.5024	0.4847	0.5052
	0.0002	**0.5520**	0.5700	0.5658	0.5662	0.5614	0.0602	0.5140	0.1353	0.5111	0.2506	0.5086	0.2506	0.5086
	0.0001	0.5766	0.5765	**0.5527**	0.5545	0.5569	0.8939	0.4990	0.0327	0.4841	0.0559	0.4857	0.1115	0.4881
$d = 2$	0.01	0.9302	0.9816	0.9365	0.9490	**0.9276**	0.0415	0.5152	0.3121	0.5075	0.2007	0.5095	0.4732	0.5053
	0.005	0.9434	0.9716	0.9521	**0.9279**	0.9456	0.2303	0.5089	0.7102	0.5028	0.9005	0.5009	0.4696	0.5054
	0.002	**0.9457**	0.9961	0.9749	0.9611	0.9859	0.0644	0.5138	0.8804	0.5011	0.3378	0.5071	0.3354	0.5072
	0.001	0.9852	0.9561	0.9783	0.9775	**0.9547**	0.2154	0.4908	0.9536	0.5004	0.9391	0.4994	0.9948	0.5000
	0.0005	0.9978	0.9784	0.9769	**0.9641**	0.9808	0.4873	0.4948	0.3322	0.4928	0.0716	0.4866	0.7090	0.4972
	0.0002	0.9871	0.9827	**0.9574**	0.9915	0.9811	0.2026	0.4905	0.1222	0.4885	0.5323	0.4953	0.3852	0.4935
	0.0001	0.9678	1.0130	0.9726	**0.9534**	0.9760	0.2234	0.5091	0.4883	0.5052	0.4522	0.4944	0.5044	0.5050
$d = 3$	0.01	0.9515	0.9639	0.9606	0.9850	**0.9491**	0.6975	0.5029	0.8514	0.5014	0.2837	0.5080	0.6560	0.4967
	0.005	0.9844	**0.9404**	0.9817	0.9803	0.9809	0.5083	0.4951	0.6035	0.5039	0.4249	0.5059	0.3965	0.5063
	0.002	0.9946	0.9853	**0.9569**	0.9918	0.9653	0.3922	0.4936	0.0275	0.4836	0.9863	0.4999	0.1312	0.4887
	0.001	0.9861	**0.9514**	0.9852	0.9757	1.0010	0.1059	0.4879	0.5522	0.5044	0.8288	0.4984	0.7370	0.5025
	0.0005	1.0068	0.9978	0.9859	**0.9510**	0.9832	0.7362	0.4975	0.4889	0.4948	0.1556	0.4894	0.2067	0.4906
	0.0002	0.9862	0.9734	0.9834	**0.9730**	0.9974	0.6592	0.4967	0.7665	0.4978	0.5442	0.4955	0.7425	0.5024
	0.0001	0.9996	**0.9945**	1.0162	1.0572	1.0066	0.9807	0.4998	0.4034	0.5062	0.0476	0.5148	0.6856	0.4970
$d = 4$	0.01	0.9984	0.9733	1.0022	0.9895	**0.9723**	0.7517	0.4976	0.5908	0.5040	0.9209	0.5007	0.4972	0.5051
	0.005	0.9913	0.9830	0.9971	0.9604	**0.9602**	0.9922	0.4999	0.7150	0.5027	0.6021	0.4961	0.3329	0.4928
	0.002	0.9949	1.0274	1.0084	**0.9749**	0.9919	0.2591	0.5084	0.6052	0.5039	0.8414	0.4985	0.7604	0.5023
	0.001	0.9839	0.9982	0.9874	**0.9767**	0.9807	0.9649	0.5003	0.9953	0.5000	0.4106	0.4939	0.9186	0.4992
	0.0005	0.9997	1.0038	1.0264	0.9968	**0.9792**	0.8139	0.5018	0.2896	0.5079	0.8391	0.4985	0.4397	0.4942
	0.0002	**0.9832**	1.0081	1.0013	1.0117	1.0206	0.1872	0.5098	0.5527	0.5044	0.0689	0.5136	0.2540	0.5085
	0.0001	**0.9693**	1.0268	0.9943	1.0038	0.9911	0.0456	0.5149	0.5598	0.5043	0.5370	0.5046	0.3980	0.5063
$d = 5$	0.01	**0.9714**	1.0162	0.9736	0.9806	1.0228	0.1548	0.5106	0.3827	0.5065	0.4749	0.5053	0.0056	0.5206
	0.005	0.9787	1.0210	1.0097	1.0002	**0.9613**	0.3629	0.5068	0.1643	0.5104	0.5924	0.5040	0.6791	0.4969
	0.002	1.0196	1.0108	0.9807	**0.9871**	1.0363	0.6384	0.5035	0.4244	0.4940	0.5290	0.4953	0.3940	0.5064
	0.001	1.0095	**0.9791**	1.0039	1.0275	1.0236	0.4323	0.4941	0.8983	0.4990	0.5204	0.5048	0.5039	0.5050
	0.0005	0.9939	1.0236	1.0298	**0.9708**	1.0223	0.3717	0.5067	0.2420	0.5087	0.7043	0.4972	0.4610	0.5055
	0.0002	0.9987	**0.9751**	0.9881	1.0208	0.9881	0.5492	0.4955	0.9281	0.5007	0.3255	0.5073	0.6977	0.5029
	0.0001	0.9954	1.0039	**0.9648**	0.9953	0.9832	0.7696	0.4978	0.4521	0.4944	0.6092	0.4962	0.4384	0.4942
$d = 8$	0.01	**0.9446**	0.9907	0.9836	0.9847	1.0045	0.0642	0.5138	0.3624	0.5068	0.2449	0.5087	0.2238	0.5091
	0.005	1.0329	1.0145	**0.9723**	1.0094	0.9781	0.3021	0.4923	0.0229	0.4830	0.6840	0.4970	0.3628	0.4932
	0.002	**0.9842**	0.9905	1.0159	1.0024	1.0316	0.3255	0.5073	0.1617	0.5104	0.6831	0.5030	0.3427	0.5071
	0.001	1.0189	1.0107	**0.9999**	1.0069	1.0411	0.4169	0.4939	0.7538	0.4977	0.6408	0.4965	0.9866	0.5001
	0.0005	1.0247	1.0123	1.0064	**0.9830**	0.9866	0.1190	0.4884	0.1358	0.4889	0.0036	0.4783	0.0290	0.4837
	0.0002	0.9792	1.0166	0.9967	**0.9596**	0.9942	0.5020	0.5050	0.8309	0.5016	0.5052	0.4950	0.6009	0.5039
	0.0001	**0.9932**	1.0207	1.0074	0.9979	0.9935	0.3650	0.5068	0.3201	0.5074	0.4655	0.5054	0.5424	0.5045
$d = 10$	0.01	**0.9760**	1.0068	1.0196	0.9771	0.9967	0.8753	0.5012	0.1357	0.5111	0.3455	0.4930	0.9348	0.4994
	0.005	1.0035	1.0265	**0.9950**	1.0067	1.0089	0.7691	0.5022	0.5492	0.5045	0.8914	0.5010	0.5979	0.4961
	0.002	0.9856	0.9933	**0.9827**	0.9882	0.9974	0.5222	0.5048	0.8152	0.4983	0.8856	0.5011	0.7253	0.5026
	0.001	1.0031	1.0083	1.0074	0.9946	**0.9834**	0.6407	0.5035	0.7539	0.5023	0.9126	0.5008	0.7832	0.4979
	0.0005	1.0161	1.0054	**1.0052**	1.0388	1.0265	0.6510	0.4966	0.4760	0.4947	0.2862	0.5079	0.4809	0.5053
	0.0002	1.0008	1.0073	1.0096	**0.9823**	1.0246	0.5084	0.5049	0.9860	0.5001	0.6003	0.4961	0.8356	0.5015
	0.0001	**0.9743**	0.9945	1.0097	0.9832	1.0149	0.2287	0.5090	0.1881	0.5098	0.9243	0.4993	0.3204	0.5074

TABLE 4
ART F-ratio results and statistical analysis comparisons among VPP-ART and other ART algorithms for *point patterns*

Dimension (d)	Failure Rate (θ)	ART F-ratio					Statistical Analysis							
							vs. FSCS-ART		*vs. Naive-KDFC*		*vs. SemiBal-KDFC*		*vs. LimBal-KDFC*	
		VPP-ART	*FSCS-ART*	*Naive-KDFC*	*SemiBal-KDFC*	*LimBal-KDFC*	*p-value*	*effect size*	*p-value*	*effect size*	*p-value*	*effect size*	*p-value*	*effect size*
	0.01	**0.9592**	0.9607	0.9755	0.9621	0.9763	0.6231	0.5037	0.4279	0.5059	0.3374	0.5072	0.0750	0.5133
	0.005	**0.9262**	0.9543	0.9563	0.9576	0.9320	0.0654	0.5137	0.0993	0.5123	0.1321	0.5112	0.6065	0.5038
	0.002	**0.9355**	0.9568	0.9627	0.9825	0.9788	0.7519	0.4976	0.8121	0.5018	0.6477	0.5034	0.1144	0.5118
$d=1$	0.001	1.0026	**0.9346**	0.9771	0.9623	0.9651	**0.0445**	0.4850	0.8614	0.4987	0.1676	0.4897	0.2653	0.4917
	0.0005	0.9779	**0.9380**	0.9815	0.9446	0.9422	0.3095	0.4924	0.5943	0.5040	0.7309	0.4974	0.2594	0.4916
	0.0002	0.9708	0.9693	0.9798	**0.9282**	0.9655	0.9067	0.4991	0.4938	0.5051	0.1021	0.4878	0.9205	0.5007
	0.0001	0.9750	0.9721	0.9807	0.9610	**0.9530**	0.8204	0.5017	0.9610	0.4996	0.9196	0.5008	0.6786	0.4969
	0.01	0.9979	0.9988	0.9918	**0.9894**	1.0207	0.2747	0.5081	0.2338	0.5089	0.8688	0.5012	0.3148	0.5075
	0.005	**0.9662**	0.9762	1.0042	0.9825	0.9917	0.8852	0.5011	0.1460	0.5108	0.5882	0.5040	0.2836	0.5080
	0.002	0.9918	0.9675	0.9718	**0.9557**	0.9877	0.7730	0.4979	0.4953	0.4949	0.1776	0.4900	0.6302	0.5036
$d=2$	0.001	0.9688	0.9995	**0.9550**	0.9672	0.9817	0.0655	0.5137	0.7319	0.4974	0.5542	0.4956	0.9392	0.4994
	0.0005	**0.9427**	0.9663	0.9650	0.9806	0.9777	0.0864	0.5128	0.1931	0.5097	**0.0339**	0.5158	**0.0245**	0.5168
	0.0002	0.9681	1.0034	0.9522	**0.9392**	0.9428	0.1068	0.5120	0.7450	0.4976	0.1525	0.4893	0.2421	0.4913
	0.0001	0.9758	0.9792	**0.9511**	0.9673	0.9556	0.9347	0.5006	0.4807	0.4947	0.8478	0.5014	0.7290	0.4974
	0.01	**1.0376**	1.1231	1.0930	1.1084	1.0795	**0.0005**	0.5260	**0.0149**	0.5182	**0.0038**	0.5216	0.1723	0.5102
	0.005	**1.0609**	1.0744	1.0973	1.0665	1.1051	0.5259	0.5047	0.0861	0.5128	0.9106	0.4992	**0.0198**	0.5174
	0.002	1.0269	**1.0235**	1.0297	1.0746	1.0499	0.7553	0.5023	0.5115	0.5049	**0.0372**	0.5155	0.1239	0.5115
$d=3$	0.001	1.0221	1.0343	**1.0151**	1.0548	1.0551	0.6355	0.5035	0.6821	0.5031	0.1631	0.5104	0.1841	0.5099
	0.0005	**0.9988**	1.0017	1.0121	1.0113	1.0077	0.8080	0.5018	0.2859	0.5080	0.3116	0.5075	0.8003	0.5019
	0.0002	1.0183	1.0093	**1.0036**	1.0122	1.0074	0.8180	0.4983	0.5799	0.4959	0.7378	0.4975	0.7097	0.4972
	0.0001	1.0023	1.0072	**0.9795**	0.9905	0.9824	0.9596	0.4996	0.2742	0.4918	0.9740	0.5002	0.5906	0.4960
	0.01	**1.2336**	1.3211	1.2789	1.3035	1.3037	**0.0023**	0.5227	0.1548	0.5106	**0.0181**	0.5176	**0.0014**	0.5238
	0.005	**1.2100**	1.2614	1.2517	1.2633	1.2192	**0.0265**	0.5165	0.2072	0.5094	**0.0424**	0.5151	0.2278	0.5090
	0.002	1.1283	1.1809	1.1735	1.1524	**1.1212**	0.1155	0.5117	**0.0355**	0.5157	0.3352	0.5072	0.8672	0.4988
$d=4$	0.001	**1.0915**	1.1137	1.1287	1.1401	1.1370	0.0935	0.5125	0.1586	0.5105	**0.0229**	0.5170	**0.0257**	0.5166
	0.0005	**1.0788**	1.1117	1.1062	1.0980	1.1065	0.1022	0.5122	0.3322	0.5072	0.2515	0.5085	0.0720	0.5134
	0.0002	**1.0444**	1.0521	1.1007	1.0487	1.0510	0.5517	0.5044	**0.0073**	0.5200	0.4581	0.5055	0.5910	0.5040
	0.0001	1.0506	1.0837	**1.0500**	1.0589	1.0509	0.2403	0.5088	0.9512	0.5005	0.6854	0.5030	0.6532	0.5033
	0.01	**1.4384**	1.5695	1.5413	1.5603	1.5243	**0.0003**	0.5273	**0.0012**	0.5241	**0.0007**	0.5253	**0.0206**	0.5173
	0.005	**1.3364**	1.4785	1.4519	1.4385	1.4456	**0.0002**	0.5278	**0.0032**	0.5219	**0.0154**	0.5181	**0.0030**	0.5222
	0.002	**1.2637**	1.3549	1.3642	1.3691	1.3510	**0.0000**	0.5332	**0.0001**	0.5288	**0.0000**	0.5324	**0.0000**	0.5365
$d=5$	0.001	**1.1862**	1.2964	1.2948	1.3005	1.2110	**0.0002**	0.5282	**0.0015**	0.5237	**0.0000**	0.5312	0.0569	0.5142
	0.0005	**1.1718**	1.2559	1.2236	1.2361	1.1938	**0.0006**	0.5256	**0.0107**	0.5190	**0.0092**	0.5194	0.1488	0.5108
	0.0002	1.1638	1.1746	1.1636	**1.1562**	1.1708	0.4006	0.5063	0.3808	0.5065	0.9149	0.4992	0.4725	0.5054
	0.0001	1.1276	1.1257	1.1474	1.1553	**1.1074**	0.6017	0.5039	0.2233	0.5091	0.1501	0.5107	0.5762	0.4958
	0.01	**2.0945**	2.4049	2.3487	2.4215	2.3374	**0.0000**	0.5377	**0.0000**	0.5366	**0.0000**	0.5400	**0.0000**	0.5339
	0.005	**2.0220**	2.3711	2.3543	2.3313	2.2386	**0.0000**	0.5443	**0.0000**	0.5396	**0.0000**	0.5387	**0.0000**	0.5333
	0.002	**1.8539**	2.1827	2.2076	2.1710	2.0722	**0.0000**	0.5515	**0.0000**	0.5491	**0.0000**	0.5432	**0.0000**	0.5380
$d=8$	0.001	**1.7151**	2.0761	2.1198	2.0804	1.9098	**0.0000**	0.5534	**0.0000**	0.5613	**0.0000**	0.5519	**0.0000**	0.5372
	0.0005	**1.6117**	1.9976	1.9948	1.9881	1.8038	**0.0000**	0.5626	**0.0000**	0.5643	**0.0000**	0.5633	**0.0000**	0.5368
	0.0002	**1.4995**	1.7979	1.7829	1.8695	1.6424	**0.0000**	0.5528	**0.0000**	0.5546	**0.0000**	0.5619	**0.0000**	0.5303
	0.0001	**1.4719**	1.7575	1.7223	1.7567	1.5938	**0.0000**	0.5539	**0.0000**	0.5450	**0.0000**	0.5512	**0.0037**	0.5216
	0.01	**2.3399**	2.5080	2.5382	2.5738	2.5994	**0.0271**	0.5165	**0.0069**	0.5201	**0.0002**	0.5277	**0.0000**	0.5320
	0.005	**2.4368**	2.8216	2.6878	2.7286	2.7150	**0.0000**	0.5437	**0.0000**	0.5313	**0.0000**	0.5428	**0.0000**	0.5379
	0.002	**2.3325**	2.8479	3.0133	2.8899	2.6569	**0.0000**	0.5561	**0.0000**	0.5735	**0.0000**	0.5577	**0.0000**	0.5387
$d=10$	0.001	**2.2444**	2.8835	2.8983	2.8322	2.6249	**0.0000**	0.5657	**0.0000**	0.5743	**0.0000**	0.5672	**0.0000**	0.5439
	0.0005	**2.0856**	2.7033	2.7247	2.6135	2.4393	**0.0000**	0.5710	**0.0000**	0.5743	**0.0000**	0.5602	**0.0000**	0.5408
	0.0002	**1.8928**	2.4606	2.4871	2.5065	2.2301	**0.0000**	0.5711	**0.0000**	0.5723	**0.0000**	0.5738	**0.0000**	0.5446
	0.0001	**1.7778**	2.0912	2.3241	2.3366	2.1341	**0.0000**	0.5445	**0.0000**	0.5708	**0.0000**	0.5744	**0.0000**	0.5553

TABLE 5
F-measure results and statistical analysis comparisons among VPP-ART and other different algorithms with the 22 subject programs

Program	d	F-measure						Statistical Analysis									
								vs. RT		*vs. FSCS-ART*		*vs. Naive-KDFC*		*vs. SemiBal-KDFC*		*vs. LimBal-KDFC*	
		VPP-ART	*RT*	*FSCS-ART*	*Naive-KDFC*	*SemiBal-KDFC*	*LimBal-KDFC*	*p-value*	*effect size*	*p-value*	*effect size*	*p-value*	*effect size*	*p-value*	*effect size*	*p-value*	*effect size*
airy	1	**797.28**	1448.70	816.03	806.91	803.29	809.33	**0.0000**	0.6248	0.1793	0.5100	0.3791	0.5066	0.4055	0.5062	0.3163	0.5075
bessj0	1	450.05	758.91	448.20	443.44	**440.31**	449.13	**0.0000**	0.5991	0.8286	0.5016	0.9175	0.5008	0.4290	0.4941	0.5561	0.5044
erfcc	1	**1019.32**	1832.56	1054.65	1040.58	1045.86	1033.00	**0.0000**	0.6067	**0.0301**	0.5162	0.3323	0.5072	0.1319	0.5112	0.2828	0.5080
probks	1	1452.91	2683.85	1469.21	**1450.82**	1452.57	1475.86	**0.0000**	0.6212	0.4970	0.5051	0.9523	0.5004	0.9427	0.5005	0.4633	0.5055
tanh	1	312.42	566.12	319.82	**306.91**	309.85	309.36	**0.0000**	0.6133	0.3084	0.5076	0.3982	0.4937	0.7942	0.4981	0.4720	0.4946
bessj	2	462.96	784.93	**452.49**	457.52	457.60	461.69	**0.0000**	0.6152	0.4585	0.4945	0.9216	0.5007	0.6456	0.4966	0.5489	0.4955
gammq	2	**1063.88**	1208.68	1087.52	1066.34	1100.74	1172.50	**0.0003**	0.5269	0.2777	0.5081	0.8889	0.5010	0.0757	0.5132	**0.0011**	0.5243
sncndn	2	640.22	**629.03**	643.40	629.63	649.75	655.74	0.9640	0.4997	0.5157	0.5048	0.9029	0.4991	0.3750	0.5066	0.1915	0.5097
golden	3	1824.80	1881.15	1831.29	**1806.09**	1816.82	1902.52	0.0562	0.5142	0.4552	0.5056	0.9594	0.4996	0.5315	0.5047	**0.0038**	0.5216
plgndr	3	1733.83	2636.11	**1572.82**	1648.26	1618.40	1665.89	**0.0000**	0.6003	**0.0076**	0.4801	0.1738	0.4899	0.0951	0.4876	0.6289	0.4964
cel	4	1628.50	3049.35	**1572.56**	1577.88	1593.71	1586.13	**0.0000**	0.6478	0.5594	0.4956	0.7333	0.4975	0.7584	0.4977	0.7521	0.4976
el2	4	796.70	1381.35	714.58	**710.58**	714.01	749.98	**0.0000**	0.6381	**0.0003**	0.4730	**0.0009**	0.4753	**0.0001**	0.4702	**0.0429**	0.4849
calDay	5	**1101.30**	1394.67	1312.37	1295.35	1262.40	1226.75	**0.0000**	0.5556	**0.0000**	0.5380	**0.0000**	0.5380	**0.0000**	0.5354	**0.0006**	0.5256
complex	6	1134.81	**1120.51**	1283.25	1155.82	1150.68	1142.01	0.3731	0.4934	**0.0002**	0.5278	0.9074	0.5009	0.8793	0.5011	0.7912	0.5020
pntLinePos	6	1397.57	**1318.12**	1589.92	1444.04	1490.97	1458.47	**0.0117**	0.4812	**0.0000**	0.5431	0.5160	0.5048	**0.0299**	0.5162	0.0980	0.5123
triangle	6	1411.86	1430.20	1396.55	1415.63	1389.04	**1324.95**	0.8492	0.5014	0.4483	0.4943	0.7890	0.4980	0.1772	0.4899	**0.0175**	0.4823
line	8	3322.27	**3178.10**	3435.07	3343.48	3269.93	3370.50	0.2682	0.4917	0.1796	0.5100	0.7428	0.5024	0.3340	0.4928	0.2840	0.5080
pntTrianglePos	8	**4584.95**	4708.16	5067.49	5046.12	4955.54	4659.86	0.9858	0.4999	0.1498	0.5107	0.2122	0.5093	0.3528	0.5069	0.3202	0.4926
TwoLinesPos	8	7415.10	**6226.52**	9814.67	8297.09	8430.90	8909.71	**0.0000**	0.4540	**0.0000**	0.5922	**0.0000**	0.5304	**0.0000**	0.5337	**0.0000**	0.5522
nearestDistance	10	**2145.31**	3964.67	2259.57	2277.88	2161.20	2188.53	**0.0000**	0.6518	**0.0000**	0.5489	**0.0176**	0.5177	0.4080	0.5062	0.1908	0.5098
calGCD	10	**1004.15**	1054.66	1016.98	1016.14	1017.74	1056.02	**0.0156**	0.5180	0.5202	0.5048	0.6552	0.5033	0.3601	0.5068	**0.0242**	0.5168
select	11	5490.70	**4759.86**	5599.86	5907.50	5634.03	5808.85	**0.0000**	0.4596	0.6502	0.5034	**0.0368**	0.5156	0.5461	0.5045	0.0885	0.5127

3.1.2 Answer to RQ1 - Part 2: Results of Experiments

Table 5 provides the detailed experimental data for the comparisons among VPP-ART and other different algorithms with the 22 subject programs.

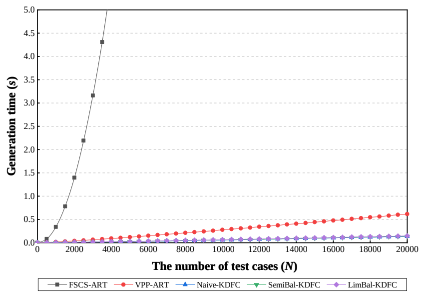
3.2 Comparisons of Efficiency

3.2.1 Answer to RQ2 - Part 1: Results of Simulations

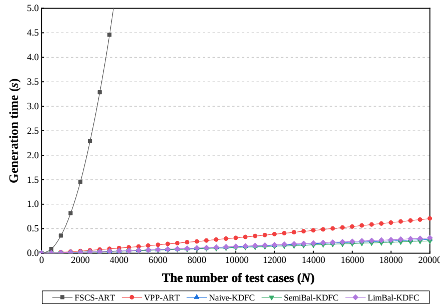
Figure 1 provides the test case generation times of VPP-ART, FSCS-ART, and the three KDFC-ART algorithms, for various

TABLE 6
F-time results and statistical analysis comparisons among VPP-ART and other algorithms with the 22 subject programs

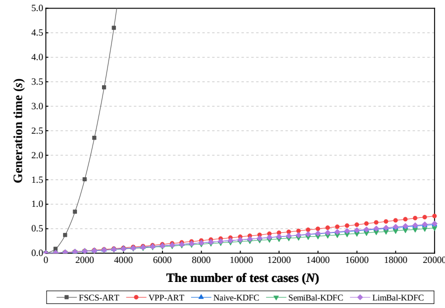
Program	d	F-time						Statistical Analysis									
								vs. RT		*vs. FSCS-ART*		*vs. Naive-KDFC*		*vs. SemiBal-KDFC*		*vs. LimBal-KDFC*	
		VPP-ART	*RT*	*FSCS-ART*	*Naive-KDFC*	*SemiBal-KDFC*	*LimBal-KDFC*	*p-value*	*effect size*	*p-value*	*effect size*	*p-value*	*effect size*	*p-value*	*effect size*	*p-value*	*effect size*
airy	1	12.74	0.34	329.08	**3.31**	3.33	3.71	**0.0000**	0.0450	**0.0000**	0.8740	**0.0000**	0.2000	**0.0000**	0.2008	**0.0000**	0.2182
bessj0	1	6.38	0.17	98.95	**1.66**	1.66	1.76	**0.0000**	0.0571	**0.0000**	0.8382	**0.0000**	0.2062	**0.0000**	0.2061	**0.0000**	0.2151
erfcc	1	17.33	0.49	529.69	**4.46**	4.60	4.80	**0.0000**	0.0371	**0.0000**	0.8904	**0.0000**	0.1919	**0.0000**	0.1971	**0.0000**	0.2039
probks	1	37.18	20.37	1081.17	18.11	**17.95**	18.57	**0.0000**	0.3139	**0.0000**	0.8845	**0.0000**	0.3000	**0.0000**	0.2973	**0.0000**	0.3057
tanh	1	3.98	0.10	50.72	**1.08**	1.17	1.21	**0.0000**	0.0683	**0.0000**	0.8273	**0.0000**	0.2044	**0.0000**	0.2173	**0.0000**	0.2218
bessj	2	7.55	0.70	132.59	8.43	**3.08**	3.97	**0.0000**	0.1331	**0.0000**	0.7889	**0.0000**	0.5326	**0.0000**	0.3214	**0.0000**	0.3647
gammq	2	20.60	0.45	775.74	16.10	**13.08**	14.28	**0.0000**	0.0582	**0.0000**	0.8319	**0.0000**	0.4520	**0.0000**	0.4153	**0.0000**	0.4330
sncndn	2	11.18	0.23	287.05	**5.53**	9.03	9.06	**0.0000**	0.0653	**0.0000**	0.7967	**0.0000**	0.3773	**0.0000**	0.4399	**0.0000**	0.4528
golden	3	45.73	3.60	2417.30	**45.16**	72.10	66.76	**0.0000**	0.1211	**0.0000**	0.8518	0.1062	0.5120	**0.0000**	0.5623	**0.0000**	0.5760
plgndr	3	42.52	7.79	1714.59	94.44	**19.43**	21.65	**0.0000**	0.2280	**0.0000**	0.8155	**0.0000**	0.6282	**0.0000**	0.3541	**0.0000**	0.3745
cel	4	36.14	0.82	1858.20	155.37	**23.46**	28.03	**0.0000**	0.0459	**0.0000**	0.8601	**0.0000**	0.7358	**0.0000**	0.4126	**0.0000**	0.4457
el2	4	**15.42**	0.46	379.12	25.32	29.34	28.23	**0.0000**	0.0703	**0.0000**	0.7910	**0.0000**	0.5933	**0.0000**	0.5869	**0.0000**	0.5968
calDay	5	**27.57**	2.19	968.30	169.88	36.91	36.18	**0.0000**	0.1083	**0.0000**	0.8371	**0.0000**	0.7733	**0.0004**	0.5264	**0.0088**	0.5195
complex	6	**27.62**	0.69	1317.92	136.16	136.54	113.95	**0.0000**	0.0525	**0.0000**	0.8536	**0.0000**	0.7358	**0.0000**	0.7446	**0.0000**	0.7431
pntLinePos	6	**34.46**	0.52	1895.50	182.67	193.30	155.95	**0.0000**	0.0396	**0.0000**	0.8563	**0.0000**	0.7496	**0.0000**	0.7640	**0.0000**	0.7651
triangle	6	**35.23**	0.61	1562.98	177.93	180.97	142.86	**0.0000**	0.0456	**0.0000**	0.8339	**0.0000**	0.7444	**0.0000**	0.7453	**0.0000**	0.7372
line	8	**102.64**	1.57	10274.08	1542.17	1659.12	936.30	**0.0000**	0.0321	**0.0000**	0.8931	**0.0000**	0.8484	**0.0000**	0.8432	**0.0000**	0.8511
pntTrianglePos	8	**153.93**	2.86	21846.93	2864.84	3077.41	1450.36	**0.0000**	0.0298	**0.0000**	0.8972	**0.0000**	0.8594	**0.0000**	0.8676	**0.0000**	0.8547
TwoLinesPos	8	**272.39**	3.27	70499.32	6015.84	6919.79	3177.67	**0.0000**	0.0222	**0.0000**	0.9347	**0.0000**	0.8831	**0.0000**	0.8946	**0.0000**	0.8870
nearestDistance	10	**69.81**	2.76	3493.54	3113.37	3467.71	1067.29	**0.0000**	0.0683	**0.0000**	0.8596	**0.0000**	0.8503	**0.0000**	0.8540	**0.0000**	0.8608
calGCD	10	**27.96**	2.48	916.10	483.69	554.96	364.48	**0.0000**	0.1273	**0.0000**	0.8139	**0.0000**	0.8033	**0.0000**	0.8119	**0.0000**	0.8055
select	11	**216.93**	5.93	26894.43	24325.93	23561.25	4338.21	**0.0000**	0.0446	**0.0000**	0.9035	**0.0000**	0.9088	**0.0000**	0.9086	**0.0000**	0.9039



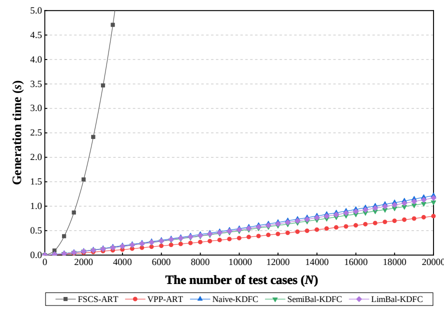
(a) $d = 1$



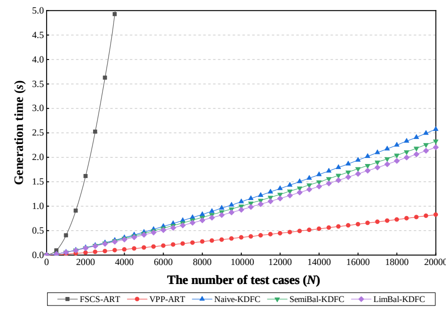
(b) $d = 2$



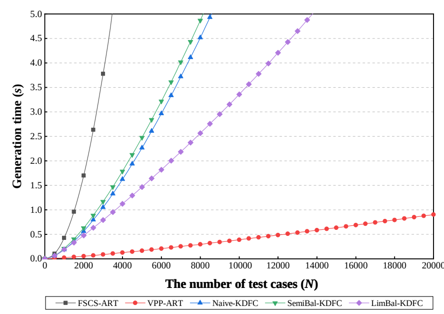
(c) $d = 3$



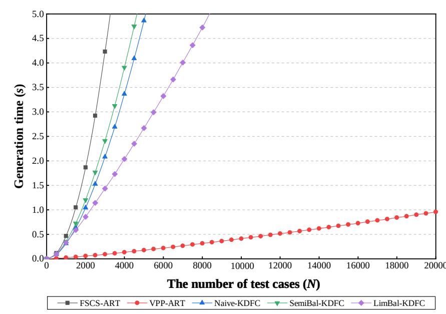
(d) $d = 4$



(e) $d = 5$



(f) $d = 8$



(g) $d = 10$

Fig. 1. Generation times for various test suite sizes in different dimensions.

test suite sizes, in different dimensional input domains ($d = 1, 2, 3, 4, 5, 8, 10$). In the figures, the x-axis shows the size of the test suite (N), and the y-axis shows the time taken to generate the N test cases.

3.2.2 Answer to RQ2 - Part 2: Results of Experiments

Table 6 reports the average time taken to detect the first failure (*F-time*) in each program, by each algorithm. In addition, this table also provides the pairwise statistical analysis of VPP-ART against other ART algorithms.