基于高斯加权的GeesePSO改进算法

庄培显 戴声奎

(华侨大学信息科学与工程学院 厦门 361021)

摘 要 为了提高粒子群算法的优化性能,通过观察和分析雁群结队飞行的智能群体现象,国内学者提出了基于雁群启示的粒子群优化算法(GeesePSO,GPSO)。该算法虽然在一定程度上提高了PSO算法的性能,但是在GPSO算法中存在着不合理的加权平均机制,即最小值寻优方面的加权缺陷。针对该问题,本文通过采用高斯加权方法对GPSO进行合理改进,提出一种基于高斯加权改进的粒子群优化算法(Gaussian-Weighted GPSO,GWGPSO)。实验结果表明:新算法在收敛精度、收敛速度和鲁棒性等指标上得到了提高,从而证明高斯加权方式是合理的和正确的。

关键词 粒子群优化,群体智能,GeesePSO,高斯加权

中图法分类号 TP181 **文献标识码** A

Improved Geese Swarm Optimization Algorithm Based on Gaussian Weighted Sum

ZHUANG Pei-xian DAI Sheng-kui

(School of Information Science and Engineering, Huaqiao University, Xiamen361021, China)

Abstract In order to improve the optimization performance for PSO, through observation and analysis of the natural phenomenon of formation flight of geese, researchers at home proposed the geese swarm optimization algorithm(GPSO). Although this algorithm has improved performance for PSO in some extent, the mechanism of average-weighted for GPSO is unreasonable, namely the defect of minimum optimization. For this issue, this paper proposed the geese swarm optimization algorithm based on gaussian-weighted(GWGPSO) through reasonable improvements of GPSO. The experimental results show that the new algorithm has improved these indicators, such as convergence precision, convergence rate and robustness, which proves that the gaussian-weighted method is reasonable and correct.

Keywords Particle swarm optimization, Swarm intelligence, GeesePSO, Gaussian-weighted

粒子群优化算法(Particle Swarm Optimization,PSO)是一种受鸟群捕食行为启示而产生的群体智能优化算法,由Kennedy和Eberhart在1995年提出和完善[1]。与其他智能算法类似,PSO是一种基于迭代的优化算法,调整参数少,易于实现。但其存在易陷入局部极值、收敛精度不高等缺点。针对这些问题,研究者们相继提出了多种改进算法。文献[2]提出线性递减惯性权重,较好地平衡了算法的局部和全局搜索能力;文献[3-7]对算法的加速度系数和边界条件等参数设置进行相关深入研究,逐步形成和完善标准粒子群优化算法(Standard Particle Swarm Optimization,SPSO)。但是在迭代初始阶段,SPSO算法计算的全局极值往往是某一个局部极值,同时由于初始阶段的收敛速度较快,因此粒子容易过快集中,导致粒子多样性不足,最终算法易陷于局部极值、收敛精度不高。为了进一步提高粒子群优化算法的性能,国内外部分学者开始将研究焦点转向生物仿生学领域,通过观察和实验发现雁群结队飞行是一种群体智能的优化行为,并且开始尝试利用雁群结队飞行的理论。文献[8]提出基于雁群启示的粒子群优化算法(Geese Particle Swarm Optimization,GeesePSO或GPSO),该算法在一定程度上提高了粒子群优化算法的收敛精度和收敛速度等性能。

在GPSO算法中,个体极值更新的加权方式仅对测试函数的最大值寻优是合理的,而对最小值寻优问题存在着不合理性。本文在GPSO基础上,提出一种基于高斯加权改进的GWGPSO算法,并通过后续的实验结果来说明新算法的收敛精度、收敛速度及鲁棒性等性能指标的优越性,从而验证GPSO算法加权方式的不合理性,进一步证明了新算法加权方式的合理性和正确性。

1 GPSO算法

近年来,为了提高粒子群算法的优化性能,国内外部分学者开始对雁群结队飞行的规律进行探索和研究。德国的空气动力学家Carl Wieselsberger首次提出了大雁飞行可以节省能量的假说。Lissaman和Schollenberger利用空气动力学理论首次给出了大雁的人字形编队能增加飞行距离的估算。借鉴雁群上述的协作飞行特性,文献[8]提出了基于雁群启示的粒子群优化算法。

GPSO算法首先随机初始化一个粒子种群M,其中,第i个粒子在N维空间中位置表示为$X_i=(x_{i1},x_{i2},\cdots,x_{iN})$,速度为$V_i=(V_{i1},V_{i2},\cdots,V_{iN})$,个体极值和全局极值分别表示为$pbest_i=(p_{i1},p_{i2},\cdots,p_{iN})$和$gbest=(gbest_1,gbest_2,\cdots,$

本文受中央高校基本科研专项(JB-ZR1145),华侨大学高层次人才科研项目(09BS102),福建省自然科学基金项目(2012J01274)资助。

庄培显(1988-),男,硕士生,主要研究方向为图像处理、智能算法,E-mail:zhuangpeixian0624@163.com;戴声奎(1971-),男,博士,副教授,硕士生导师,主要研究方向为图像处理、模式识别、智能算法。

$gbest_N$)。在GPSO算法中，将粒子看作大雁，首先将粒子按照适应度好坏依次排序，并将适应度最好的粒子作为头雁。排序后，将前一个粒子的个体极值 p_{i-1} 作为当前粒子的全局极值 $gbest$。另一方面，所有大雁根据个体极值 $pbest_i$ 和适应度值 $f(X_i)$ 进行加权平均，并将所有大雁的个体极值更新为此加权平均值。GPSO算法的全局极值、个体极值、速度和位置更新分别如下：

$$p_g = p_{i-1} \tag{1}$$

$$p_a = \frac{\sum_{i=1}^{M} pbest_i \times f(X_i)}{\sum_{i=1}^{M} f(X_i)} \tag{2}$$

$$V_i(k+1) = w(k) \times V_i(k) + c_1 \times r_1 \times (p_a(k) - X_i(k)) + c_2 \times r_2 \times (p_{i-1}(k) - X_i(k)) \tag{3}$$

$$X_i(k+1) = X_i(k) + V_i(k+1) \tag{4}$$

$$w(k) = w_{\max} - \frac{(w_{\max} - w_{\min}) \times k}{k_{\max}} \tag{5}$$

式中，$V_i(k)$，$X_i(k)$ 分别为第 i 个粒子的速度和位置，k 为当前迭代次数，$k_{\max}$ 为最大迭代次数。r_1 和 r_2 是服从[0,1]均匀分布的随机数，c_1 和 c_2 为非负的加速度学习因子，$w_{\max}$ 和 $w_{\min}$ 分别为 w 的最大值和最小值，式(5)是 Shi. Y 和 Eberhart 提出的权重 w 线性递减方法，在一定程度上调整了算法的全局和局部的搜索能力[2]。

经实验证实，GPSO算法在一定程度上提高了算法的收敛精度，同时收敛速度和算法的鲁棒性也得到了提高[8,9]。但是式(2)形式意味着每只大雁都知道所有大雁的状态，通过后续的测试实验发现：由于直接用函数的适应度值作为权重，该加权平均方式对测试函数的最大值寻优是有利的，而对最小值寻优上却存在着严重的问题，在最小值寻优后期，个体极值逐渐接近最优解，权重越来越小，个体极值与较小的权重相乘，不利于迭代后期的全局极值向着最优解收敛。并且证实了基于式(2)的改进实际上起反作用，因此GPSO算法的加权平均方式在函数最小值寻优方面还存在着很大的改进空间。

2 GWGPSO算法

2.1 GWGPSO算法原理

为了有效地解决GSPO算法中加权平均所带来的问题，受高斯函数特性研究的启示，提出了高斯加权平均的改进方法。在最小值寻优的迭代后期，个体极值逐渐逼近最优解，通过高斯函数加权，距离最优解较近的个体极值对应较大的权重，粒子的个体极值与较大的权重相乘，利于全局极值向着最优解的方向收敛。

本文对GPSO做出如下改进：GWGPSO算法仍将粒子看作飞行的大雁，将按照粒子的适应度好坏依次排序当前粒子，并将粒子群中适应度最好的粒子作为头雁。首先，利用所有粒子的适应度值$\{f(X_i), i=1,2,\cdots,M\}$来计算高斯加权时的均值 $E[f(X_i)]$和方差 σ^2，如式(6)和式(7)所示，即

$$E[f(X_i)] = \frac{1}{M}\sum_{i=1}^{M} f(X_i) \tag{6}$$

$$\sigma^2 = \frac{1}{M-1}\sum_{i=1}^{M} (f(X_i) - E[f(X_i)])^2 \tag{7}$$

然后，比较和计算得到当前粒子群的最小适应度值 $fitness\_min$ 并将其作为后续高斯加权的中心，通过适应度的高斯函数计算来更新当前粒子的加权权重 $w(X_i)$，即

$$w(X_i) = \exp(-(f(X_i) - fitness\_min)^2/(2\times\sigma^2)) \tag{8}$$

最后，将所有大雁根据个体极值 $pbest_i$ 和适应度值 $w(X_i)$ 进行加权平均，并将此值作为所有大雁的全局极值，即

$$p_{ga} = \frac{\sum_{i=1}^{M} pbest_i \times w(X_i)}{\sum_{i=1}^{M} w(X_i)} \tag{9}$$

因此，GWGSPO算法的速度更新如下：

$$V_i(k+1) = w(k) \times V_i(k) + c_1 \times r_1 \times (p_{i-1}(k) - X_i(k)) + c_2 \times r_2 \times (p_{ga}(k) - X_i(k)) \tag{10}$$

则粒子的位置更新与式(4)相同。

2.2 GWGPSO算法实现

GWGPSO算法实现步骤如下(见图1)。

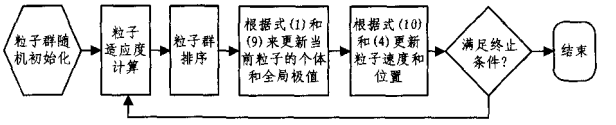

图1 GWGPSO算法流程图

1)粒子种群随机初始化：初始化粒子种群、初始位置及速度等实验参数。2)通过利用测试函数来计算每个粒子的适应度值。3)粒子群排序：选出最优适应度最好的粒子作为头雁，按照适应度值的好坏依次排序粒子群。4)个体极值和全局极值更新：计算式(1)和式(9)来分别更新当前粒子的个体极值和全局极值。5)粒子位置和速度更新：根据式(10)和式(4)来更新当前粒子的速度和位置。6)判断是否满足终止条件，如果满足，则算法迭代结束，否则，转入步骤2)进行迭代循环。

3 实验结果及分析

为了验证和对比新算法的性能，本文选取3个典型的Benchmark函数进行测试[8,9]，这些函数的理论最优值全为0。其中，Sphere和Rosenbrock函数在平坦区域中具有单个极值，而Rastrigrin函数具有多个极值。每个函数独立实验50次，每次内部迭代1000次，种群规模选择为20，w 线性下降变化范围为[0.9,0.4]，c_1 和 c_2 均为2.0，$V_{\max}$ 根据函数的特点按实际情况选取。Benchmark测试函数的参数及初始范围如表1所列。

表1 Benchmark测试函数及初始化参数

测试函数	解析式	维数	初始范围
Sphere	$f_1(X) = \sum_{i=1}^{N} x_i^2$	30	[−10,10]
Rosenbrock	$f_2(X) = \sum_{i=1}^{N-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	10	[−10,10]
Rastrigrin	$f_3(X) = \sum_{i=1}^{N} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	[−5.2,5.2]
Schaffer	$f_4(X) = \frac{(\sin\sqrt{\sum_{i=1}^{N} x_i^2})^2 - 0.5}{[1 + 0.001(\sum_{i=1}^{N} x_i^2)]^2} + 0.5$	20	[−10,10]

3.1 固定维数和迭代次数时算法测试

给定的维数和迭代次数采用上述的初始参数。采用最优适应度的平均最优解、最优解及方差3个评价指标来评价新算法的性能。其中，最优适应度的平均最优解及最优解用来衡量算法的收敛精度、反映算法的全局寻优能力，标准差用于衡量算法的鲁棒性。优异比率为50次实验中GPSO和GWGPSO分别相对SPSO时较优实验次数的比例。表2为固定维数和迭代次数的实验结果。

表 2　固定维数和迭代次数的实验结果

测试函数	优化算法	平均最优解	最优解	标准差	优异比率
Sphere	SPSO	0.0069	0.0026	0.0038	
	GPSO	0.0023	0.0009	0.0008	98%
	GWGPO	0.0017	0.0008	0.0005	100%
Rosenbrock	SPSO	42.9084	25.4906	27.7592	
	GPSO	28.2095	24.4970	1.3014	84%
	GWGPO	28.0529	23.1444	1.2863	86%
Rastrigrin	SPSO	34.9237	17.4732	9.1296	
	GPSO	11.8868	7.4056	3.3287	100%
	GWGPO	10.7512	4.7884	2.7788	100%
Schaffer	SPSO	0.06359	0.03722	0.02195	
	GPSO	0.03776	0.00971	0.00995	62%
	GWGPO	0.03694	0.00971	0.00806	64%

由表 2 的实验数据可得：GWGPSO 平均最优解、最优解及标准差最小，说明本文算法具有更高的收敛精度、更强的全局寻优能力和更好的稳定性。在 50 次实验中，GWGPSO 算法在统计较优实验次数上优于 GPSO 算法。

3.2　函数维数变化时性能测试

为了测试本文算法在测试函数维数变化时的性能，选择 Rastrigrin 函数为基准测试函数，维数由 10 逐步增加到 30，其余参数不变。不同维数下的实验结果如表 3 所列。

表 3　不同维数下的实验结果

维数	优化算法	平均最优解	最优解	标准差	优异比率
10	SPSO	11.3436	2.9849	4.2217	
	GPSO	4.3388	0.9951	2.0476	96%
	GWGPSO	3.5424	0.0003	1.3951	98%
20	SPSO	23.5562	9.5176	7.9729	
	GPSO	8.0122	5.0148	2.4099	100%
	GWGPSO	7.8407	2.2111	2.2281	100%
30	SPSO	34.9237	17.4732	9.1296	
	GPSO	11.8868	7.4056	3.3287	100%
	GWGPSO	10.7512	4.7884	2.7788	100%

随着 Rastrigrin 函数的维数变化，GWGPSO 的平均最优解、最优解及标准差都是最小的，该算法的收敛精度、全局寻优能力及鲁棒性等指标在统计意义上是领先的。

3.3　算法收敛速度和收敛趋势对比

为了观察算法的收敛速度和收敛趋势，将算法迭代过程中全局最优值（即最优个体适应度）以曲线的方式进行显示。试验时迭代次数为 1000，粒子维数为 30，其余试验参数不变。3 种算法迭代时的曲线如图 2 所示（其中，曲线的纵坐标代表全局极值的适应度值，横坐标代表迭代次数，GANGPSO 为 GWGPSO）。

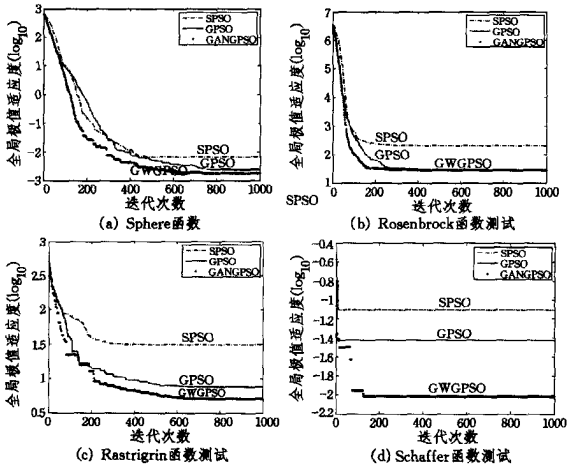

图 2　3 种算法的适应度曲线图

从图 2 所示可知，随着迭代次数增加，GWGPSO 适应度曲线始终处在 SPSO 和 GPSO 的下方，表明 GWGPSO 的收敛趋势和收敛精度好于前两种算法。在设定相同的收敛精度时，GWGPSO 适应度曲线对应的迭代次数最小，说明 GWGPSO 的收敛速度快于 SPSO 和 GPSO。

3.4　规则 1 和规则 2 的性能试验

为了更好验证 GSPO 算法的加权问题，规则 1：GPSO 和 GWGPSO 算法只修改全局极值部分；规则 2：GPSO 和 GWGPSO 算法全局极值更新与 SPSO 相同；其余部分与 SPSO 算法相同。单独使用规则 1 和规则 2 的算法性能，分别对 Rastrigrin 函数作多次试验，采用默认参数，得到的试验结果如表 4 所列。

表 4　规则 1 和规则 2 的影响试验

规则	优化算法	平均最优解	最优解	标准差	优异比率
仅规则 1	SPSO	36.7234	18.9574	11.1750	
	GPSO	54.2055	19.9266	26.8304	24%
	GWGPSO	21.2701	11.8062	6.7473	96%
仅规则 2	SPSO	36.2616	18.3955	8.3081	
	GPSO	21.6986	9.3170	8.1647	96%
	GWGPSO	21.5985	7.2605	6.8838	96%

通过表 4 数据可知：仅采用规则 1 时，GPSO 算法的结果明显比 SPSO 差，GPSO 的优异比率结果明显变差，而仅采用规则 2 时，GPSO 算法要明显优于 SPSO。说明了规则 1 实际上对 GPSO 算法在极值寻优方面存在反作用，其原因正如本文在第 1 部分指出的 GPSO 算法缺陷。而在上述两种情况下，GWGPSO 算法均优于 GPSO 和 SPSO，从而证明高斯加权方式的合理性和有效性。

3.5　算法时间复杂度的对比

为了比较 3 种算法的时间复杂度，采用算法运算时间来衡量。每个函数进行 50 次实验，每次实验运行 1000 次迭代，利用 50 次实验的总时间的平均值来计算每次实验时间，如表 5 所列。

表 5　算法的时间复杂度比较

测试函数	优化算法	运算时间(S)
Sphere	SPSO	0.779592
	GPSO	0.870499
	GWGPSO	0.986402
Rosenbrock	SPSO	0.781911
	GPSO	0.806826
	GWGPSO	0.874536
Rastrigrin	SPSO	0.756018
	GPSO	0.890184
	GWGPSO	0.954947
Schaffer	SPSO	0.801444
	GPSO	0.840953
	GWGPSO	0.910339

从表 5 数据统计意义上可得 GWGPSO 与 SPSO、GPSO 这两种算法的运行时间相差不大，GWGPSO 牺牲了部分运算量来获得提高算法的收敛性能。但是从算法最终的折中效果可得出新算法具有可行性。

结束语　针对 GPSO 算法中加权方式在最小值寻优方面存在的问题，本文对此进行改进，提出了基于高斯加权改进的 GeesePSO 算法（GWGPSO），并通过实验证明：新算法具有更高的收敛精度、更快的收敛速度及更好的鲁棒性等指标，验证

（下转第 124 页）

结束语 本文针对当前大规模地形绘制遇到的大数据传输的问题，提出一种支持大规模地形数据的动态调度和绘制算法。利用 GPU 的实时计算和 CPU 的动态分析，在绘制每一帧时只载入当前渲染所需的极少部分瓦片，并利用地理数据的特性简化了地形块的管理和采样。算法简化了数据的交换并且改进了内存的管理，能够有效减少宽带的开销，对于大数据的传输具有很好的执行效率。目前该算法的改进方向是把更多的计算任务交由 GPU 来完成，以释放更多的 CPU 资源。我们下一步工作的重点一是把瓦片的管理移植到 GPU 进行，二是利用 GPU 完成瓦片的选取。

参 考 文 献

[1] Duchaineauy M. ROAMing Terrain: Real-time Optimally Adapting Meshes[C]// Proceedings of IEEE Visualization'97. 1997: 81-88

[2] Rottger S. Real-time generation of continuous levels of detail for height fields [C]// Proceedings of W-SCG'98. 1998: 315-322

[3] Turner B. Real-Time Dynamic Level of Detail Terrain Rendering with ROAM[EB/OL]. http://www.gamasutra.com/features/20000403/turner_01.htm, 2007-04

[4] Pajarola R. Large Scale Terrain Visualization Using the Restricted Quadtree Triangulation[C]// Proceedings of IEEE Visualization'98. 1998: 19-26

[5] Lindstrom P, Pascucci V. Terrain Simplification Simplified: A General Framework for View-Dependent Out-of-Core Visualization [M]. IEEE Transaction on Visualization and Computer Graphics. 2002

[6] 戴晨光，张永生，邓雪清. 海量地形数据实时可视化算法[J]. 计算机辅助设计与图形学，2004，16(11)：1603-1607

[7] 李胜，冀俊峰，刘学慧，等. 超大规模地形场景的高性能漫游[J]. 软件学报，2006，17(03)：535-545

[8] Li L, Li F, Huang T. Smooth Schedule of Large-Scale Terrain Visualization from External Memory[J]. Journal of Software. 2007, 18(sup): 26-34

[9] de Boer W H. Fast Terrain Rendering Using Geometrical Mipmapping[EB/OL]. http://www.connectii.net/emersion, 2000

[10] Tanner C C, Migdal C J, Jones M T. The Clipmap: A Virtual Miamap[C]// Proceedings of the ACM SIGGRAPH'04. 1998: 151-158

[11] Losasso F, Hoppe H. Geometry clipmaps: Terrain rendering using nested regular grids [J]. ACM Trans on Graphics, 2004, 23(3): 769-776

[12] Asirvatham A, Hoppe H. Terrain rendering using GPU-based geometry clipmaps, GPU Gems 2: Programming Techniques for High-Performance Graphics and General Purpose Computation [M]. Boston, MA: Addison-Wesley Professional, 2005: 14-30

[13] Foundation W. MegaTextures[EB/OL]. Retrieved October 10, from http://en.wikipedia.org/wiki/megatextures, 2006

[14] Barrett S. Sparse virtual textures [EB/OL]. Game Developer Conference, San Francisco, CA. http://silverspaceship.com/src/svt, 2008

[15] Neu A. Virtual texturing[D]. CoRR, abs/1005.3163, 2010

[16] van Waveren J M P. id tech 5 challenges-from texture virtualization to massive parallelization[EB/OL]. http://s09.idav.ucdavis.edu/talks/05-JP_id_Tech_5_Challenges.pdf, 2009

[17] Hoppe H. Smooth view-dependent level-of-detail control and its application to terrain rendering [C]// Proceedings of Conference on Visualization. Los Alamitos: IEEE Computer Society Press, 1998: 35-42

[18] 白皓，龚光红，丁莹. 基于 Clipmap 的大规模地形可视化技术研究[J]. 中国体视学与图像分析，2009，14(2)：202-208

(上接第 89 页)

了本算法的高斯加权方式的合理性和正确性。但是高斯加权只是一种简单的加权方式，还可能会存在更好的加权方法，并且引入高斯加权改进后增加了算法的时间复杂度，因此下一步还需要对新算法进行更深入的探索和研究，以更好地解决新算法的时间复杂度和加权改进方式等问题。

参 考 文 献

[1] Kennedy J, Eberhart R C. Particle Swarm Optimization [A]// Proceedings of the IEEE International Conference on Neural Networks [C]. 1995: 1942-1948

[2] Eberhart R C, Kennedy J. A New Optimizer Using Particle Swarm Theory [C]// Sixth International Symposium on Micro Machine and Human Science. 1995: 39-43

[3] Shi Y H, Eberhart R C. Empirical Study of Particle Swarm Optimization [A]// Proceeding of Congress on Evolutionary Computation [C]. Piscataway, NJ: IEEE Service Center, 1999: 1945-1949

[4] Shi Y H, Eberhart R C. A Modified Particle Swarm Optimizer [C]// IEEE International Conference on Evolutionary Computation. Anchorage, Alaska, May 1998: 69-73

[5] Ratnaweera A, Halgamuge S. Self-organizing hierarchical particle swarm optimizer with time 2 varying acceleration coefficients [J]. IEEE Trans Evolutionary Computation, 2004, 8(3): 240-255

[6] Robinson A, Rahamat-Samii Y. Particle Swarm Optimization in Electromagnetics[J]. IEEE Trans. Antennas Propag., 2004: 397-407

[7] Shi Y H, Eberhart R C. Parameter Selection in Particle Swarm Optimization[C]// Annual Conference on Evolutionary Programming. San Diego, March 1998: 591-600

[8] Liu Jin-yang, Guo M Z, Deng C. GeesePSO: An Efficient Improvement to Particle Swarm Optimization[J]. Computer Science, 2006, 33(11): 166-168

[9] Xiao Z, Yuan Y, Li P Y. Learning Algorithm for Multimodal Optimization[C]// Proceedings of the ELSEVIER International Conference on Computer and Mathematics with Applications. Zhengzhou, China, June 2009: 2016-2021