PLUNIFY

InTime Default 配方与布局扫种子的时序性能对比

01 摘要

这篇白皮书对比了两种时序优化方法的效果。一种是使用来自 FPGA 设计优化工具 InTime 的 Default 配方，另一种是我们所熟知的“扫种子(Seed Sweep)”。

InTime Default 配方使用机器学习来为设计寻找优良的综合和布局布线的参数组合。它可以共享对不同设计的深入分析，从而发挥可靠的效果。

扫种子会改变 Quartus 的 Fitter 种子值，这个值对设计的初始布局设置有影响。而改变种子值会在布局布线的开始阶段就改变一个设计的条件，从而导致 Fitter 结果的波动。这个方法虽然是全世界的设计团队都会使用的设计优化方法，但是效果的随机性很强，并且没有一个“黄金种子值”可以应用到所有的设计中。

本次实验使用一个 Stratix V 的设计和一个 Arria 10的设计。对每个设计， InTime Default 配方会编译200个数据点，扫种子编译200个数据点。我们将会比较最差余量(WS)，总负余量(TNS)，最大频率(Fmax)和运行时间的最佳值。

对于 Stratix V 的设计，InTime Default 配方将最差余量提升了57.38%，但扫种子仅仅提升了13.46%。

对于 Arria 10的设计，InTime Default 配方将最差余量提升了34.98%，但扫种子仅仅提升了11.44%。

由此可见，InTime Default配方在时序性能方面比扫种子表现更加出色。

02 设计细节

下面两个表格将详细介绍本次实验中的两个设计。

表2.1: Stratix V设计

设计信息	
Quartus Prime版本	17.1.0 Build 590 10/25/2017 SJ Standard Edition
版本名称	jesdsv
Top-level Entity 名称	jesdrx
系列	**Stratix V**
器件	**5SGSMD3H3F35I4**
时序信息	
Clock name	rxlink_clk
Corner	Slow 850mV -40C Mode
最差余量, WS (ns)	-0.765
总负余量, TNS (ns)	-897.292
最大频率，Fmax (MHz)	306.28
利用率信息	
Logic utilization (in ALMs)	42,553 / 89,000 (48 %)
Total registers	47330
Total pins	138 / 544 (25 %)
Total virtual pins	6,540
Total block memory bits	81,920 / 14,090,240 (< 1 %)
Total DSP Blocks	0 / 600 (0 %)
Total PLLs	0 / 52 (0 %)
Total DLLs	0 / 4 (0 %)

表2.2: Arria 10设计

设计信息	
Quartus Prime版本	17.1.0 Build 240 10/25/2017 SJ Pro Edition
版本名称	jesda 10
Top-level Entity 名称	jesdrx
系列	**Arria 10**
器件	**10AX027E4F27E3LG**

时序信息	
Clock name	rxlink_clk
Corner	Slow 900mV 100C Model
最差余量, WS (ns)	-0.769
总负余量, TNS (ns)	-977.742
最大频率，Fmax (MHz)	305.90
利用率信息	
Logic utilization (in ALMs)	52,860 / 101,620 (52 %)
Total registers	56480
Total pins	66 / 296 (22 %)
Total virtual pins	8,262
Total block memory bits	102,400 / 15,360,000 (< 1 %)
Total DSP Blocks	0 / 830 (0 %)
Total HSSI RX channels	0 / 12 (0 %)
Total HSSI TX channels	0 / 12 (0 %)
Total PLLs	0 / 32 (0 %)

03 测试步骤

实验按下列步骤进行：

1. 用打开Stratix V的设计。
2. 运行InTime Default配方20轮，每一轮运行20次编译。
3. 对200次编译运行扫种子，每一次使用不同的种子值。
4. 对比最差余量(WN)，总负余量(TNS)和最大频率(Fmax)。
5. 对Arria 10的设计重复1-4步。

04 结果

表4.1和4.2通过最差余量WS和总负余量TNS来对比两种优化方法对Stratix V和Arria 10设计的优化。

表 4.1: 最差余量 WS 对比

最差余量, WS (ns)					
	原始值 (ns)	扫种子		InTime Default 配方	
		余量 (ns)	提升(%)	余量 (ns)	提升(%)
Stratix V 设计	-0.765	-0.662	13.46	-0.326	57.38
Arria 10 设计	-0.769	-0.681	11.44	-0.5	34.98

表4.2: 总负余量 TNS 对比

最佳总负余量, TNS (ns)					
	原始值 (ns)	扫种子		InTime Default 配方	
		余量 (ns)	提升(%)	余量 (ns)	提升(%)
Stratix V 设计	-897.292	-745.228	16.95	-39.736	95.57
Arria 10 设计	-977.742	-635.613	34.99	-75.44	92.28

表4.3: 最大频率 Fmax 对比

最大频率, Fmax (MHz)					
	原始值 (MHz)	扫种子		InTime Default 配方	
		最大频率 (MHz)	提升(%)	最大频率 (MHz)	提升(%)
Stratix V 设计	306.28	316.25	3.26	353.89	15.54
Arria 10 设计	305.90	314.37	2.76	333.33	8.97

05 优化流程和运行时间

对于扫种子，流程很简单，只需把"Fitter 种子"从1到200设置一遍。默认值是1。对于使用InTime Default 配方，我们会运行10轮，每轮20次编译。每一轮在上一轮结束之后开始。因此，两种方法的运行时间会有所不同。

下图5.1显示的是运行时间的不同。值得一提的是InTime Default配方的平均运行时间比扫种子要高一些。另外，因为InTime Default配方比扫种子尝试了更加多样的参数设置，所以这种方法的编译时间波动会比较大。又因为InTime Default配方需要在每轮结束之后分析结果，并由此产生下一轮的设置，所以该方法的总运行时间也会略长。

表5.1：不同并行运行和轮数的运行时间比较

	运行时间 (小时)					
	原始运行时间 (小时)	并行运行数	扫种子 (1轮，200个编译)		InTime Default 配方 (10轮，每轮20个编译)	
			总运行时间 (小时)	平均运行时间 (小时)	总运行时间 (小时)	平均运行时间 (小时)
Stratix V 设计	0.5	5	26	0.6	37	0.7
Arria 10 设计	0.5	3	37	0.8	38	1.0

05.1 Stratix V 项目结果

以下是Stratix V项目的结果。

05.1.1 种子轮

图 5.1.1:200 个种子的总负余量 TNS(ns)

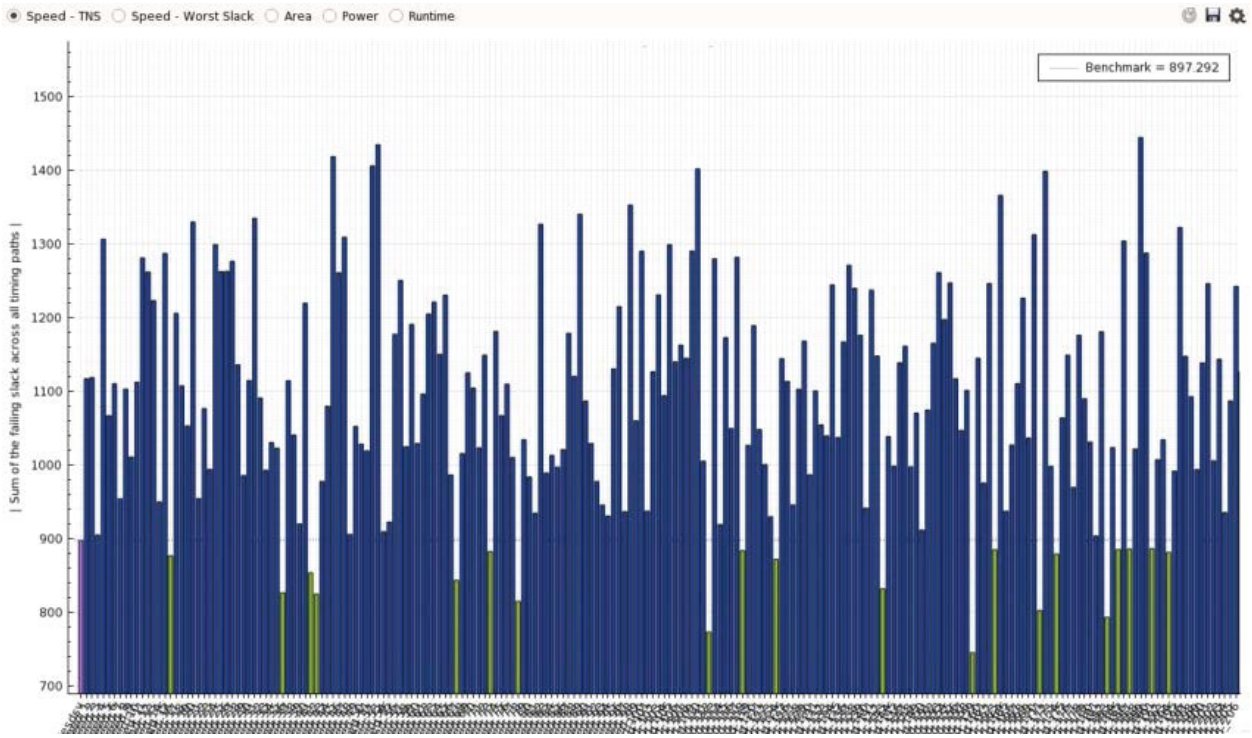
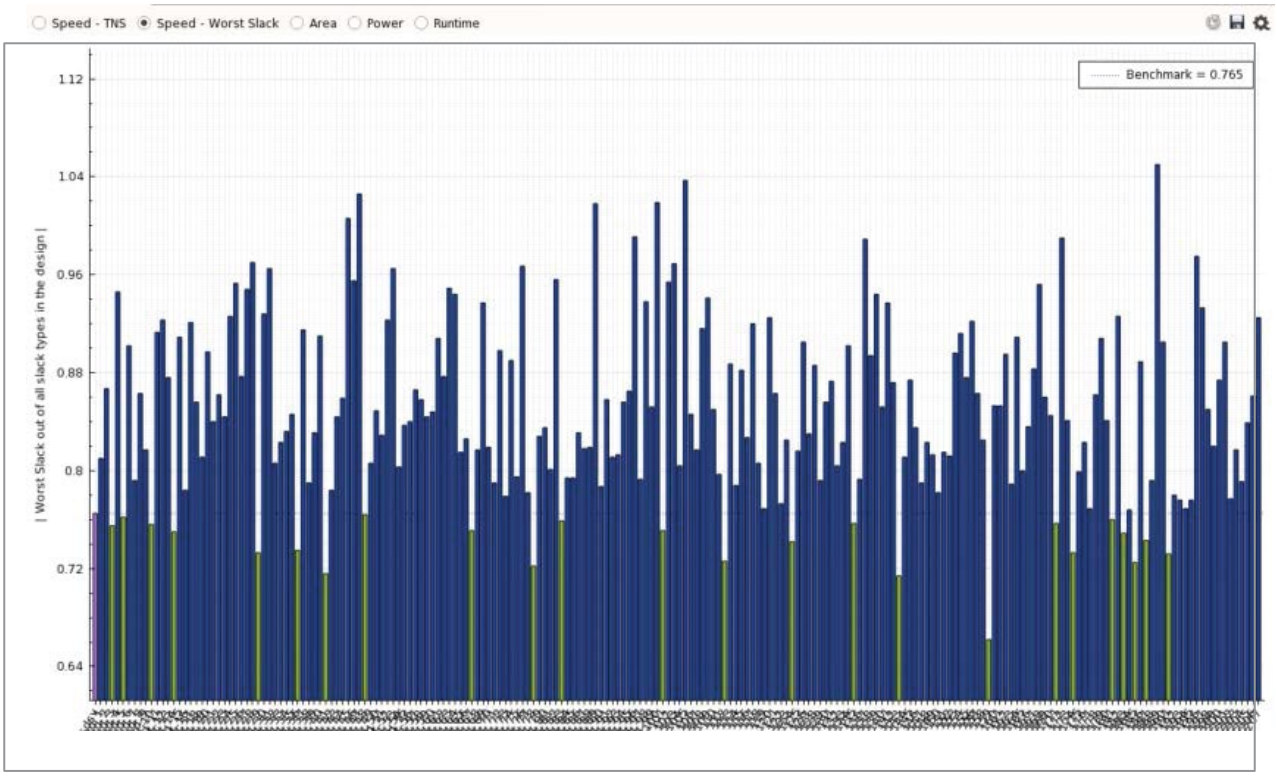


图 5.1.2:200 个种子的最差余量 WS(ns)



05.1.2 InTime Default 配方

图 5.1.3: InTime Default配方的总负余量 TNS(ns)

(请注意Y轴因为波动较大，以指数表示。)

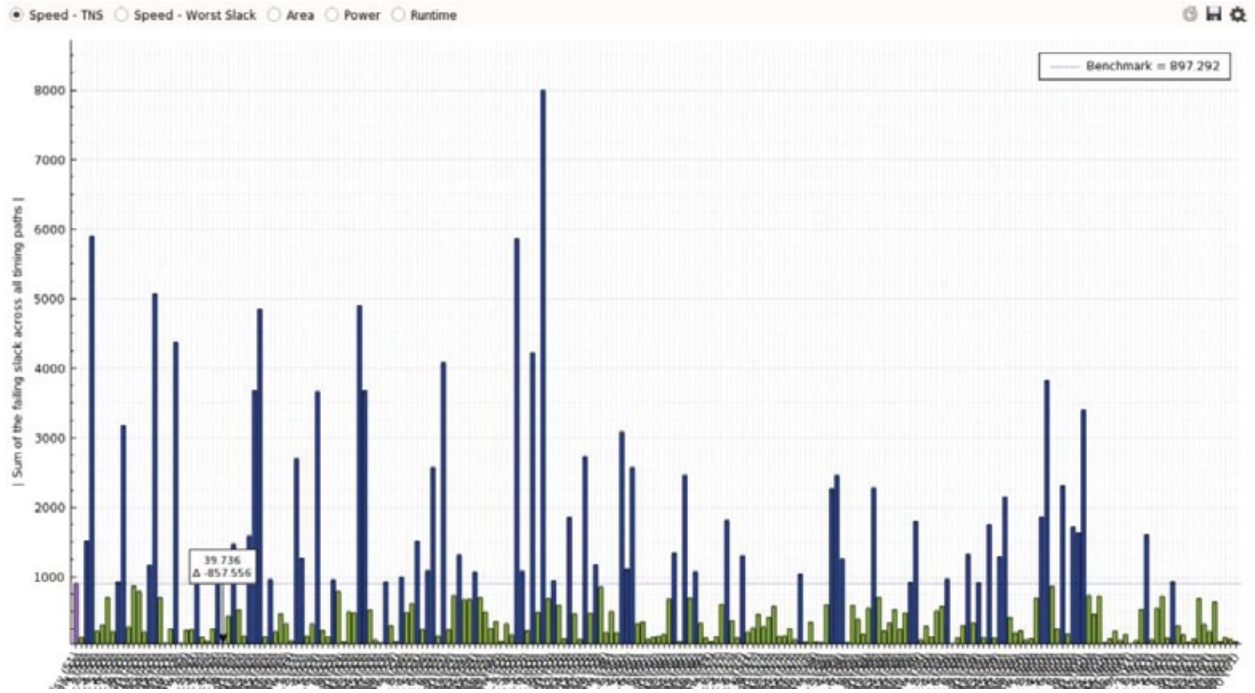
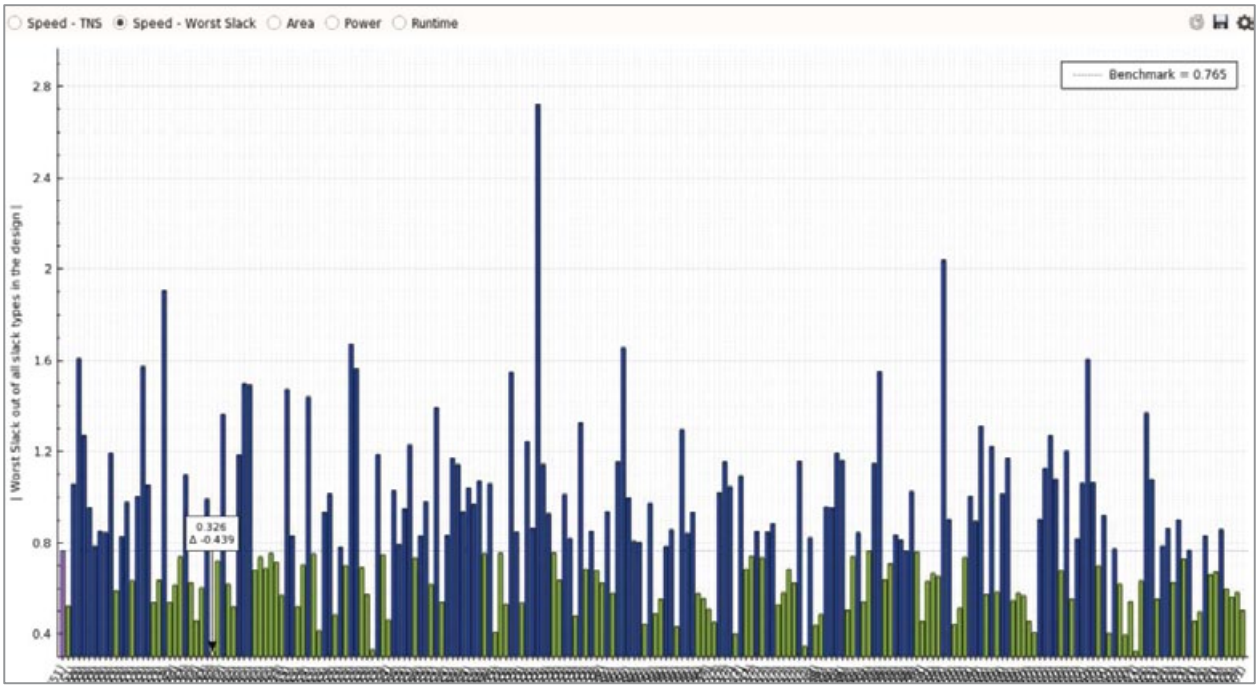


图 5.1.4: InTime Default配方的最差余量 WS(ns)



下图展示的是每一轮过后结果是如何提升的。X轴表示的是任务，每一栏代表一个任务。绿线是每个任务中最佳的结果，红线是最差的。

图5.1.5: InTime Default配方的总负余量值 TNS

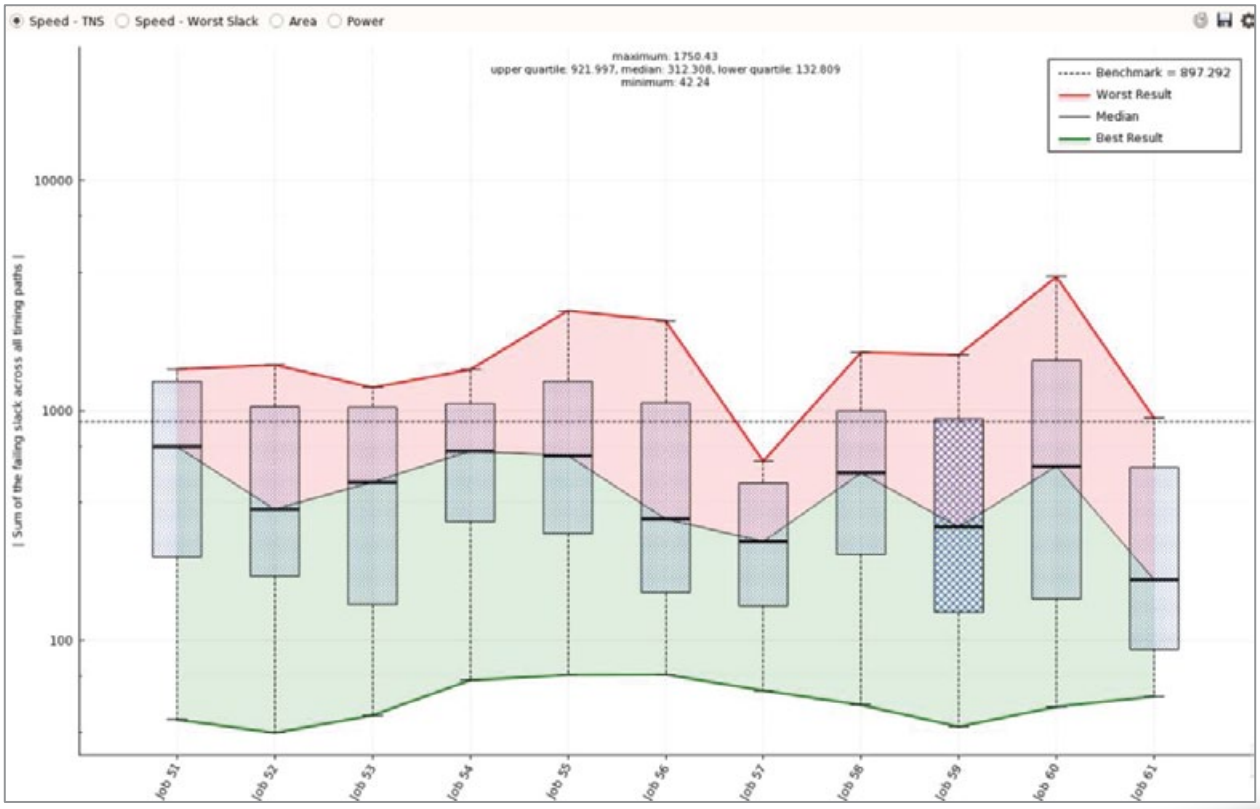


图5.1.6: InTime Default配方的最差余量值 （ WS ）

05.2 Arria 10 项目结果

以下是 Arria 10项目的结果。

05.2.1 种子轮

下图展示的是1到200种子值的结果。绿色的结果优于原始结果，蓝色的结果差于原始结果。

图5.2.1: 200个种子的总负余量值 TNS(ns)

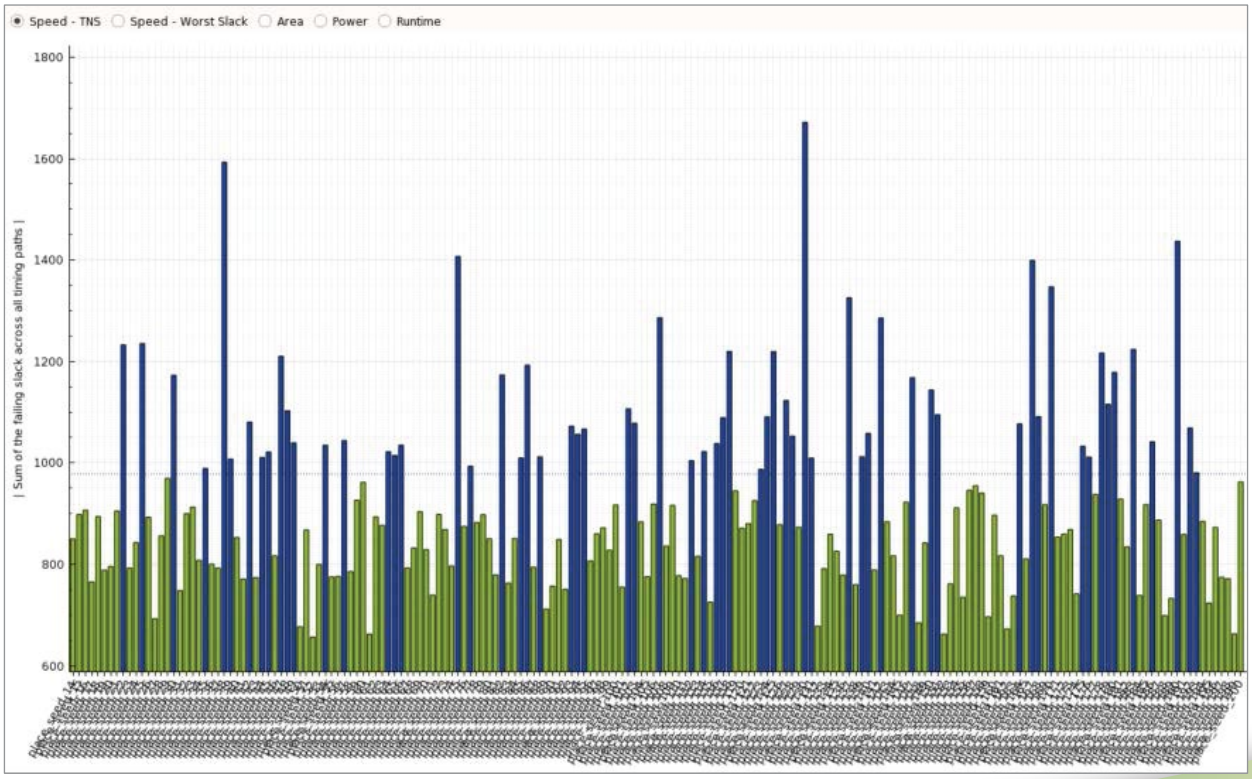
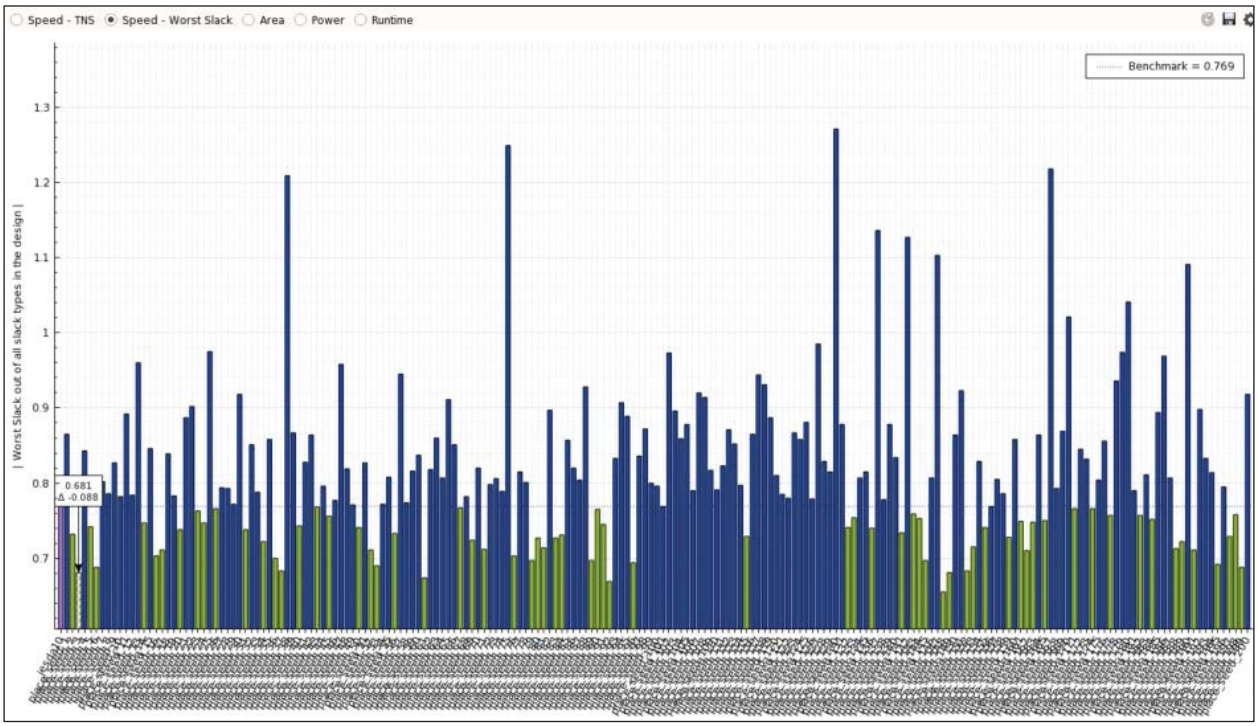


图5.2.2: 200个种子的最差余量值 WS(ns)



05.2.2 InTime Default 配方

InTime 每轮运行20个编译，运行10轮。每轮结束之后，InTime分析数据，并从中"学习"以生成下一轮的参数。

图 5.2.3: InTime Default配方的总负余量值 TNS

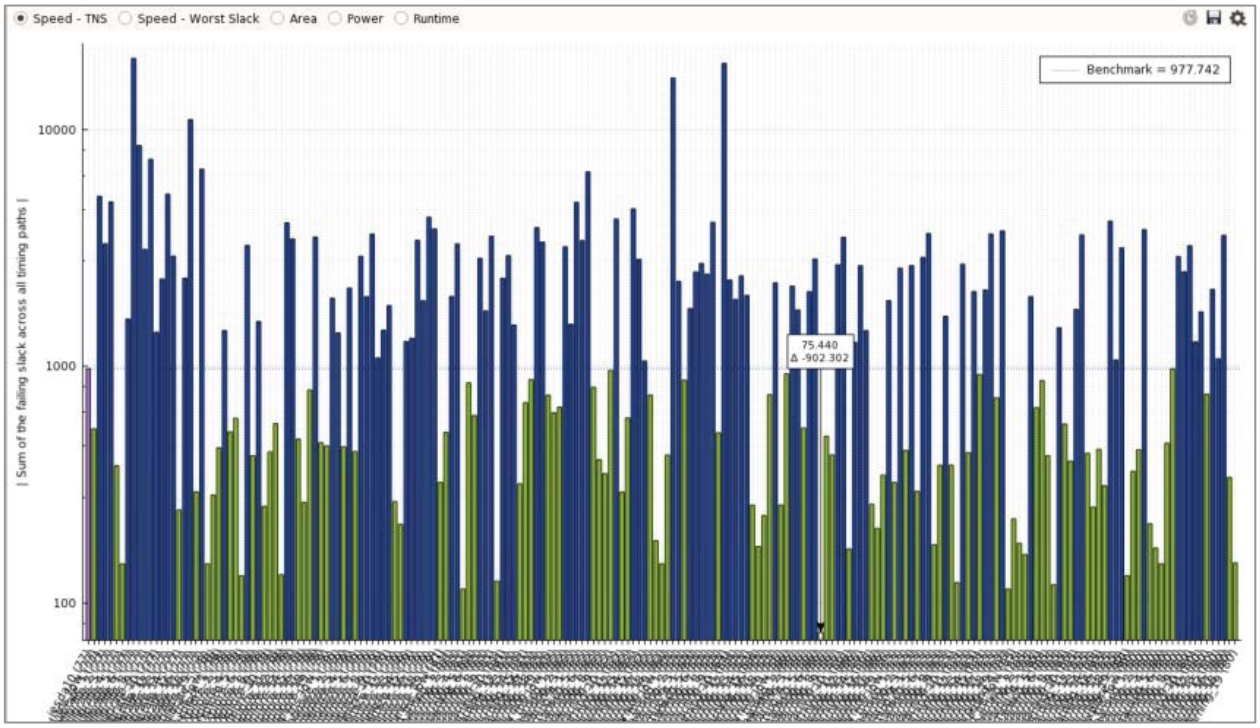
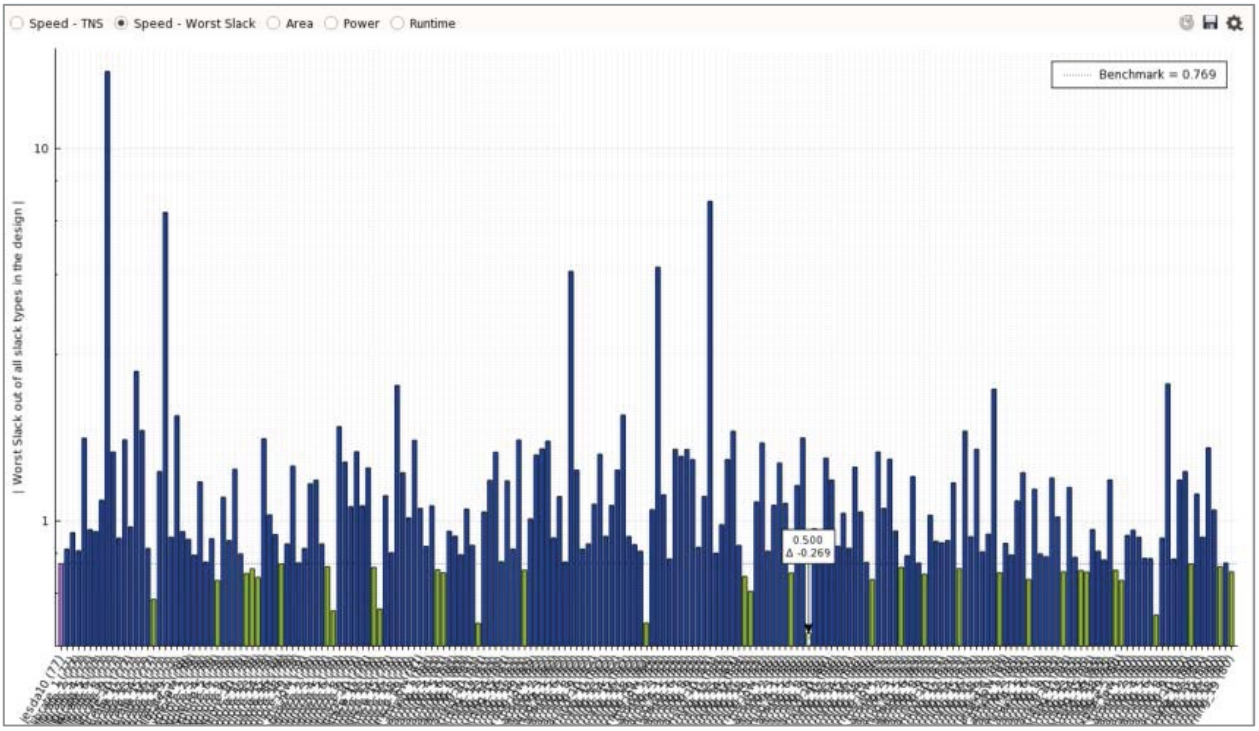


图 5.2.4: InTime Default 配方的最差余量值 WS



下图展示的是每一轮过后结果是如何提升的。X轴表示的是任务，每一栏代表一个任务。绿线是每个任务中最佳的结果，红线是最差的。

图5.2.5: InTime Default配方的总负余量值 TNS 提升（10轮）

图5.2.6: InTime Default 配方的最差余量值 WS提升（10轮）

06 结论

在实现更高的时序性能方面，InTime更加有效。在相同数量的编译下（200次），InTime Default配方可以实现30%-50%的性能提升，而使用fitter种子，只能提升11%-14%。

而且，如果将两种方法结合使用，也就是在良好的 InTime Default 配方结果之上使用扫种子，性能提升会更加明显。InTime工具就是通过这个流程来提升时序表现并缩短上市时间的。关键就在于在扫种子之前首先要取得一个足够好的结果，这样效率就大大提升了。

想要了解更多关于InTime和它的功能，请移步至https://www.plunify/com/cn/intime/