表 6-25 未被杀死变异体数量及对应的变异算子统计结果

实验对象	变异算子	未被杀死变异体数目	占总体比例
BillCalculation	AOIS	12	40.00%
	AORB	8	26.67%
	ODL	2	6.67%
	SDL	8	26.67%
	VDL	2	6.67%
AirlinesBaggageBillingService	AOIS	1	10.00%
	AORB	4	40.00%
	CDL	3	30.00%
	SDL	3	30.00%
ExpenseReimbursementSystem	AOIS	15	71.43%
	ROR	5	23.81%
	SDL	1	4.76%
MealOrderingSystem	AOIS	14	32.56%
	AORB	15	34.88%
	CDL	6	13.95%
	ODL	6	13.95%
	SDL	2	4.65%

从表格中不难发现，变异算子 AOIS 产生的变异体占所有未被杀死变异体的比重最大。那么这究竟是为什么呢？

本研究选取 MealOrderingSystem 未被杀死变异体中的 AOIS_70 为例子，说明为什么 METRIC*识别的蜕变关系比较难杀死 AOIS 产生的变异体。

AOIS_70 的变异位置是 MealOrderingSystem 源代码中第 91 行，将

this.msr.numberOfChildMeals = this.numberOfChildPassengers * 2 变异为

this.msr.numberOfChildMeals = --this.numberOfChildPassengers * 2，即在 this.numberOfChildPassengers 前加了“--”，让该变量先自减 1 再参与运算。NumberOfChildMeals 是 MealOrderingSystem 的一个输出，这个变异体的植入故障的位置恰好位于输出的前一个环节。将 this.numberOfChildPassengers 作为自变量，this.msr.numberOfChildMeals 作为因变量，研究错误植入前后自变量与因变

量之间关系的变化。若用x代表自变量，y代表因变量，则植入故障前因变量与自变量间的关系是：

$$y = x * 2,$$

植入故障后因变量与自变量之间的关系是：

$$y = (x - 1) * 2,$$

将自变量与因变量之间的关系使用函数图像表示，如图 6-21 和 6-22 所示：

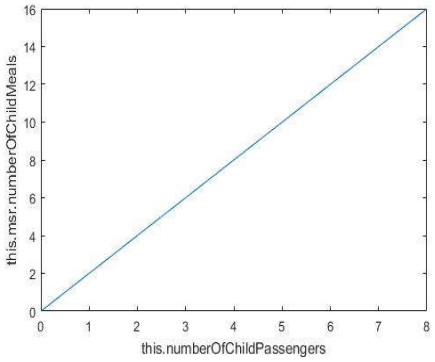

图 6-21 变异前的函数图像

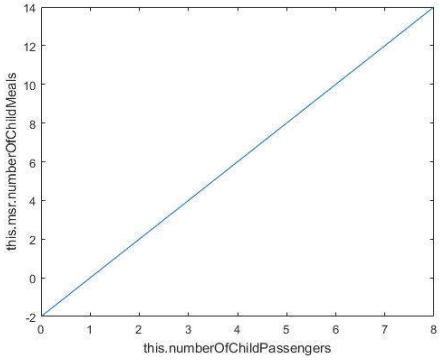

图 6-22 变异后的函数图像

通过两个图我们可以发现植入的故障仅仅将函数图像向下平移了两个单位距离，并没有改变函数的图像。蜕变测试通过比较多个输入对应的输出之间是否满足一定的蜕变关系来检测故障。具体到这个例子中，实际上就是比较函数图像上多个点之间是否满足一定的变化关系。例子中的故障仅仅是将这个函数进行了平移，并没有能力改变点与点之间的变化关系。或者说，该故障的植入并没有导致函数的趋势发生变化，从而比较多个点之间是否满足一定的变化关系就不能检测出该故障了。

在此，本研究的一个猜想是：由 METRIC*识别的蜕变关系较难检出那些不能够改变程序输出变化趋势的变异算子所代表的故障。更一般的一个猜想是蜕变测试较难检出那些不能够改变程序输出变化趋势的变异算子所代表的故障。

6.6 总结

该部分对前文的研究问题进行回答，并总结实验中的不足之处。

1. **满足测试帧覆盖的蜕变关系集的故障检测能力怎样？**

   答：测试用例集的变异得分最低为 68.60%，最高为 91.67%。此外，研究发现提升测试用例集的规模能够提升测试用例集的故障检测能力，但是