THE STUDY OF SCP OVERLOAD CONTROL IN MULTI-SERVICE ENVIRONMENT*

Li Tonghong  Liao Jianxin  Chen Junliang
(National Key Laboratory of Switching Technology and Telecommunication Networks,
Beijing University of Posts and Telecommunications, Beijing 100876)

Abstract  This paper first gives a SCP abstract model, then SCP's overload detection and maximum processing capability are discussed quantitatively. Based upon dynamic adjustment, a new two-level SCP overload control algorithm is proposed. Theoretical analysis and simulation prove the algorithm's effectiveness and fairness.

Key words  IN (intelligent network), Multi-service; Overload detection; Dynamic adjustment; Two-level control algorithm

I. Introduction

The purpose of Intelligent Network (IN) is to rapidly and efficiently supply new services that meet various customers' need. In order to achieve it, control is separated from switch in IN's architecture, Service Switching Point (SSP) only performs basic switching function, Service Logic Program (SLP) and subscriber data required by IN services are distributed in Service Control Point(SCP). SCP controls service's execution through interpreting SLP, which is downloaded to SCP by Service Management System(SMS). Generally, there exist various SLPs in each SCP, which accordingly supply many kinds of services.

If the arrival rate of calls to SCP is high enough to exceed SCP's processing capability, SCP will be overloaded. The number of successfully completed services (throughput) will be dropped dramatically unless SCP limits the offered service rates from the SSPs. There are many papers[1-6] about IN's overload, but all of them assume SCP only implements a kind of service, in which environment SCP's service processing time and the maximum processing capability are fixed, and overload's detection and control are very simple. But in multi-service environment, since each service requires a different processing, the average processing time per call changes according to the traffic pattern, and the maximum number of service processed by SCP is not fixed, which can be explained by following example.

Suppose the SCP handles service A, B and C, and that the processing time for each service is 1, 5 and 20ms. Under traffic pattern X, let the offered rate of service A be 80 calls/s, that of service B be 10 calls/s and that of service C be 10 calls/s. That is, the offered load to the SCP is 0.33, and the SCP can accept 100 calls/s. On the other hand, under traffic pattern Y, let the offered rate of service A be 10 calls/s, that of service B be 10 calls/s and that of service C be 80 calls/s. That is, the total offered rate is 100 calls/s, the same rate as traffic pattern X. However, the offered load to the SCP is more than 1, so

*Supported by 863 high technology project, the National Natural Science Foundation of China and Key Foundation of Ministry of Posts and Telecommunications
the SCP cannot accept 100 calls/s.

IN's overload control must fulfill following objectives: (1) Effectiveness. That is to say, maximize SCP's throughput in event of congestion. In multiservice environment, what represents SCP's maximum processing capability is a problem worth discussing. From the above example, we see it cannot be represented by the maximum number of services processed by SCP per unit time, it must be redefined. (2) Fairness. In multiservice environment, it must assure services of a type with a low offered rate are accepted without regulation. Service provider can define different services' accepted ratios in event of congestion according to their processing times and revenues. (3) Robustness. The algorithm must be robust to different overload patterns, to customer behavior (e.g. customer reattempt). (4) The algorithm must be easy to implement.

This paper is organized as follows: Section II presents SCP's abstract model, SCP's overload detection and maximum processing capability are discussed quantitatively. Section III provides SCP's overload control frame, and proposes a overload control algorithm A, which is two-level control algorithm based on dynamic adjustment. Section IV proves algorithm A satisfies our defined effectiveness and fairness. The result of simulation is given in Section V.

II. SCP's Overload Detection and Maximum Throughput

1. SCP's processing model

For any SCP, there exist four different user processes no matter how it is implemented: RECEIVE (receiving messages), SEND (transmitting messages), MAIN (main control) and SP(interpreting SLP). RECEIVE has the highest priority. When the SCP receives messages from IN's other nodes every interrupt, RECEIVE puts corresponding messages into receiving buffer. MAIN executes following steps repetitively:

(1) check up whether receiving buffer is empty or not. If not, put decoded messages into message queue.

(2) get message from the head of queue, if the message has waited for time \(T\) in the queue (\(T\) is subscriber's maximum waiting time, it is generally 1 or 2 s.) , it is discarded. Otherwise, different branches are executed according to message type. If the message should be transmitted to other nodes goto (3); otherwise goto (4). If queue is empty, goto (1).

(3) call SEND, then goto (1).

(4) call SP, SP invokes corresponding SLP according to message type. In the process of interpreting SLP, SP will generate different messages, which are put into message queue. Each service's execution can be regarded as a finite state machine. It is always in some states, waiting certain type of message. After receiving, SP will process it and enter new state. When SP is finished, goto (1).

2. IN's overload detection and SCP's maximum throughput

Each process must occupy CPU resource when executing. If the SCP's load is small, that is to say, there are not many messages waiting in the queue, then no messages are discarded. With the increase of offered service rate, SP's CPU occupancy, the number of messages in the queue and message's waiting time will be increased. When the offered